UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA PARA REUTILIZACIÓN DE TROQUELES MEDIANTE RAZONAMIENTO BASADO EN CASOS

T E S I S

QUE PARA OBTENER EL TÍTULO DE
INGENIERO EN COMPUTACIÓN

P R E S E N T A:

ISMAEL VELASCO POCEROS

DIRECTORES DE TESIS:
DR. VÍCTOR HUGO JACOBO ARMENDÁRIZ
ING. MANUEL AUGUSTO MANRÍQUEZ MIRANDA

CD. UNIVERSITARIA ABRIL 2008

Titulo: Sistema para reutilización de troqueles mediante razonamiento basado en casos

Índice

Capítulo 1 Planteamiento del problema.

1.1 Descripción de la empresa

Grupo Zapata, es una empresa con 80 años dedicada a proveer una amplia variedad de soluciones de envasado tanto en formas, medidas y usos. En sus orígenes comenzó fabricando productos de limpieza y aseo para el calzado, a la cual se incorporó al poco tiempo la fabricación de envases de hoja de lata.

Para la fabricación de estos productos la empresa utiliza diversas máquinas, herramientas y troqueles, estos últimos son los que ayudan en mayor medida a la producción.

Durante todos estos años, la empresa ha generado una gran cantidad de troqueles con los que se han realizado diversidad de envases, estas herramientas no han tenido un adecuado control y almacenamiento, lo que ha llevado a no tener con claridad especificaciones que permitan su reutilización en la creación de productos de un nuevo pedido.

1.2 Administración de las herramientas

Se llevaron a cabo visitas a la empresa para conocer el manejo de las herramientas, el fin principal fue revisar documentación y registros de troqueles, para conocer la forma en la que el personal que tiene contacto con los troqueles los identifica, con esto la primera observación fue que un troquel podría no tener registro, que lo tiene o tener dos o más.

Los números que puede tener un troquel son asignados por el plano con el que se diseñó pero ciertamente existen herramientas que por su tiempo se ha perdido información. Otra numeración es asignada de acuerdo a un listado consecutivo de las herramientas que se usan constantemente y se encuentran en almacén, subalmacenes, en mantenimiento y líneas de producción.

Herramientas que no fueron diseñadas dentro de la empresa presentan otro registro que se queda aunado al que la empresa le asigna o la que su plano indica, es por esto que un troquel puede presentar varios números diferentes.

En ocasiones la experiencia de los trabajadores es la que permite localizar una herramienta, sin embargo son muy pocos los que pueden hacerlo y les lleva varios minutos encontrarla.

Se ubicaron diversas áreas en donde se almacenan troqueles. Las que se identificaron fueron: almacén general, subalmacenes y algunas otras no definidas como almacén pero que se han utilizado para éste fin. Muchas herramientas no están dentro de estos sitios, simplemente se encuentran regados en la planta sin registro ni manera de clasificarlos y otras más, por su uso constante, no es posible desmontarlos de las máquinas, por lo tanto su clasificación también es difícil conocerla.

1.2.1 Definición del problema

El problema a resolver consiste en encontrar troqueles que sean reutilizables, es decir emplear uno o varios de los troqueles que se encuentran en la planta, en la creación de nuevos productos.

Todos los números que puede tener un troquel no da los datos necesarios para una identificación rápida y adecuada, al no tener una clasificación estándar esto dificulta su reutilización ya que no hay forma exacta de saber para cual producto fue empleado a menos de que sea de un uso constante. El no poder identificar un troquel con facilidad se traduce en tiempo y dinero perdidos para la creación de un nuevo producto que se le solicita a la empresa.

Los productos están determinados por múltiples características requeridas por los clientes, estas características están dadas por materiales, herramientas y diferentes técnicas de manufactura, por lo que es importante saber en que productos se emplearon los troqueles.

En resumen, se necesita contar con una clasificación que proporcione datos acerca del troquel, tener una forma rápida y adecuada de almacenar y recuperar la información, así como encontrar características relevantes que ayuden en la fabricación de un producto nuevo teniendo siempre en cuenta los troqueles con los que se fabrica.

1.3 Formulación de la propuesta

Este proyecto esta orientado a establecer acciones que permitan a la empresa hacer más eficientes sus procesos de diseño, manufactura y administración de troqueles. En específico, se plantea lo siguiente:

El primer paso es crear un código de identificación de los troqueles que estará basado en las características más significativas que los definen, también formarán parte elementos que eviten la repetición de códigos. Su conformación deberá ser fácil de entender de acuerdo a sus dígitos y no será extenso

Teniendo ya definidos los elementos que conformarán el código, el troquel será identificado por éste y se podrá almacenar en una base de datos, la entrada y salida de datos se hace a través de un sistema de cómputo para aprovechar mejor la información almacenada. Los usuarios del sistema podrán realizar el levantamiento de la información, modificarla, darla de baja y hacer consultas de ella según lo que requieran saber.

Ya que el propósito principal del sistema es obtener información que permita tomar herramientas para reutilizarlas en nuevos productos, se manejará la técnica de Razonamiento Basado en Casos (RBC).

El empleo de esta técnica se debe a que proporciona una manera de solucionar un problema utilizando las características que lo definen, haciendo la comparación de estas características con las de problemas o situaciones anteriores, se obtendrán soluciones aproximadas o similares que ayuden a cumplir el propósito principal.

El problema es encontrar troqueles que se usaron anteriormente en productos que la empresa elabora y que pueden servir en un nuevo pedido, por eso es indispensable una técnica como la de RBC donde es posible usar las características para encontrar productos similares con su solución, que son los troqueles con los que fue realizado.

En la siguiente figura se muestra la ruta que el usuario seguirá ayudado por el sistema, para conseguir el o los troqueles que le pueden servir en la creación de un nuevo producto.

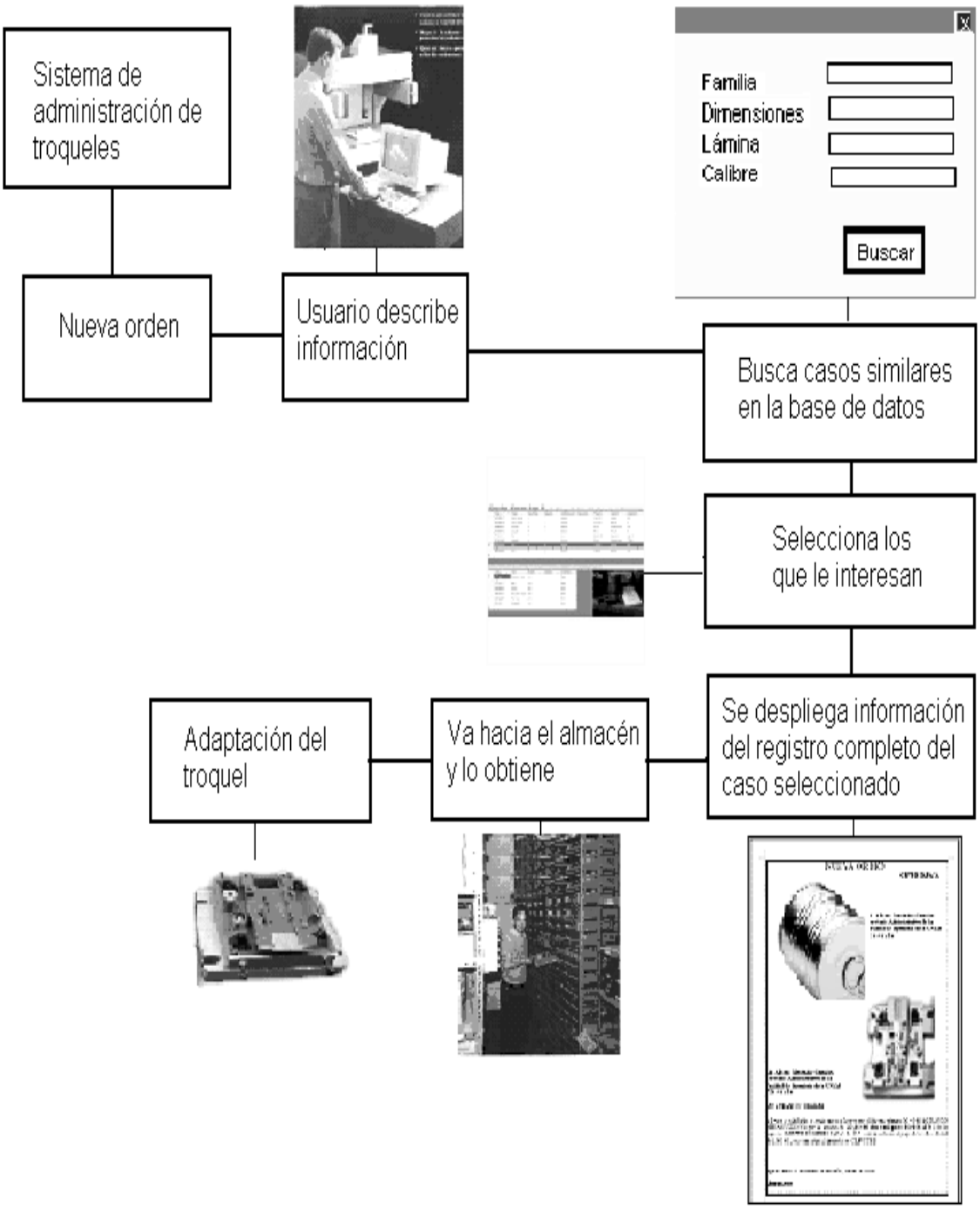


Figura 1. Dinámica de operación del sistema de administración de troqueles

1.3.1 Justificación y relevancia

La implementación de un sistema de cómputo permite que el personal relacionado con la producción pueda tener una herramienta de información con la cual ahorrar tiempo y dinero. En este momento la localización de los troqueles depende mucho de un experto, que pese a su experiencia de varios años, el tiempo en la búsqueda suele ser tardado.

Es relevante ya que actualmente la empresa no cuenta con ningún sistema de cómputo y base de datos donde se puedan consultar las características que ayuden a encontrar e identificar troqueles reutilizables.

Al definir un espacio permanente para almacenar las herramientas, los encargados podrán hacer un levantamiento de información más rápido, determinarán una posición en los anaqueles y localizarlos en menos tiempo.
Los posibles usuarios que tendrán acceso al sistema son personal del área de ventas, los cuales son los encargados de recibir los pedidos de un nuevo producto, el área de ingeniería que son los encargados de fabricar el troquel y el personal del almacén que es donde las herramientas estarán resguardadas, por lo que todos podrán hacer uso de la misma información.

1.4 limitaciones

Hay acciones en las que es importante el criterio humano relacionado con la adaptación de los troqueles.

El sistema no es una herramienta de diseño para nuevos productos, ni dará soluciones exactas a menos de que todas las características concuerden con las de un producto anterior almacenado en la base de casos.

Los errores en la información dependerá del personal que la capture, el sistema únicamente hará la validación del tipo de datos.

Capítulo 2 Troqueles.

2.1 Troquel.

Un troquel (figura 2), es una herramienta que generalmente esta constituido por un punzón y una matriz, se monta a una prensa o algún otro equipo en el que se coloca el punzón a una parte móvil y la matriz se mantiene fija en su contraparte, mediante unos postes guía aseguraran el alineamiento entre las dos partes.

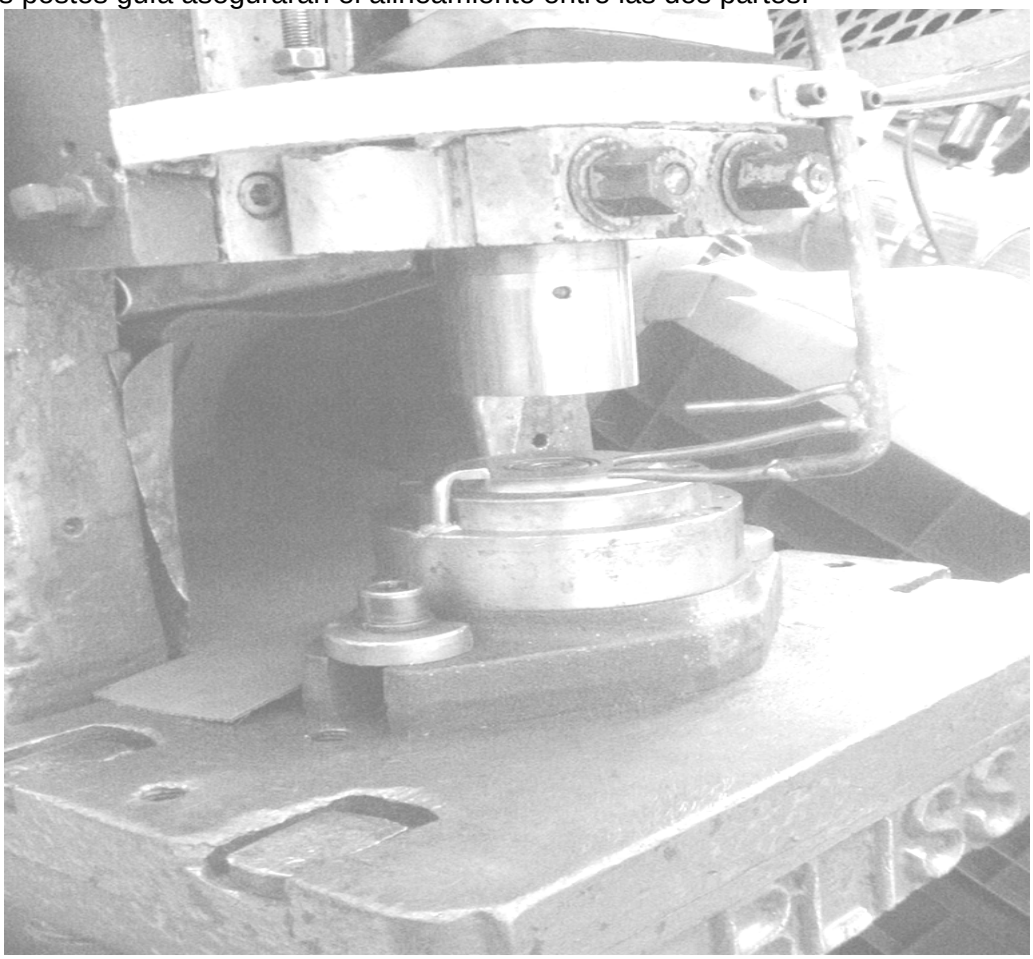


Figura 2. Troquel

Su finalidad es hacer un producto de materiales como la lámina, el plástico, cuero, goma, etc., utilizando diferentes técnicas de manufactura, la matriz y el punzón le dan la forma.

En su fabricación un producto pasa por varias operaciones o etapas en las que se le da forma a sus componentes, cuerpo, tapa, fondo, etc., de acuerdo al diseño se pueden utilizar varios troqueles, cada operación puede ser hecha por uno, pero también existen troqueles que combinan las técnicas para poder realizar un producto en un menor número de operaciones.

2.2 Tipos de Troqueles.

Los troqueles se distinguen por el tipo de operación que realizan, siendo las tres principales, la de corte, la de doblado y la de embutido. Dentro de cada una existen otras técnicas que tienen como principio alguna de estas tres.

Troquel de punzonado o corte.

El corte significa separar una pieza de una cinta con un corte a lo largo de una sola línea. La partición significa que se quita desperdicio entre las dos piezas para separarlas.
Estos troqueles emplean la técnica de punzonado que consiste en obtener una figura determinada sobre una lámina o una placa, cortándola mediante el punzón y la matriz.
Se utiliza para la fabricación de grandes cantidades de piezas, por lo tanto, la reducción de los desperdicios de material representa un factor importante para la producción económica. Las dimensiones de la matriz y punzón están determinadas por el tamaño y la forma de la pieza que se quiere obtener. En la figura 3 se muestra un troquel de punzonado.

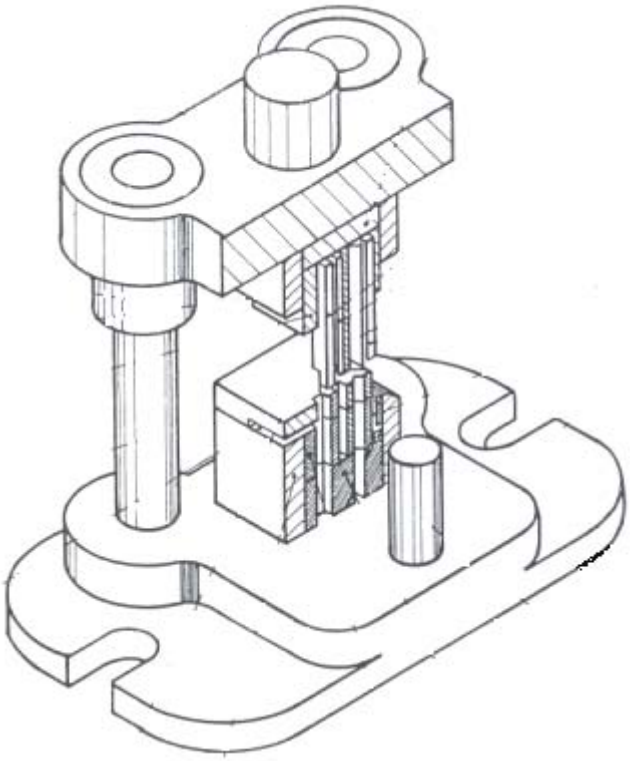


Figura 3. Troquel de punzonado.

Troquel de doblado.

Realizan el doblado o plegado que consiste en variar la forma de una lámina, manteniendo el paralelismo de sus caras y el espesor.
El doblado se realiza entre ángulos que varían de 0^o a 180^o con punzones redondeados, para evitar que la chapa se rompa durante la operación.

El doblado simple consiste en un dobles recto a la hoja de metal; operaciones tales como la formación de rebordes, formación de juntas y el plegado son similares aunque un poco más complicadas.
El radio mínimo del doblez, varia de acuerdo con la ductibilidad y espesor del metal, aunque es recomendable que no sea menor al valor del espesor del material.

Los troqueles para doblado se pueden dividir en tres grupos generales:

El primer grupo, comprende únicamente las herramientas que reciben la pieza previamente punzonada. Su misión es efectuar los dobleces, curvaturas, etc., que den el acabado final a la pieza de trabajo. . La figura 4 es un dibujo de un troquel de doblado del primer grupo.

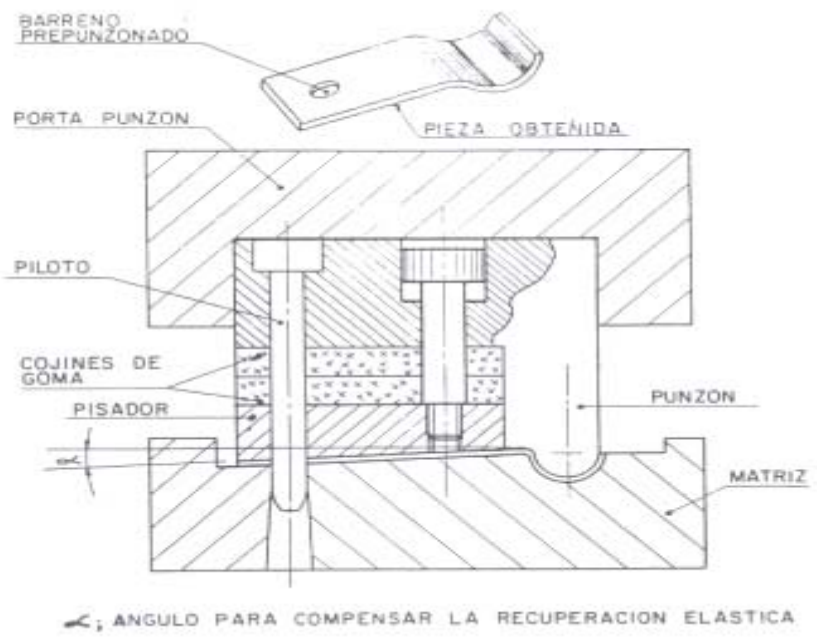


Figura 4. Troquel de doblado, primer grupo

El segundo grupo lo forman los troqueles que punzonan y doblan; proporcionando la pieza terminada. Un dibujo de este tipo de troquel se ve en la figura 5.

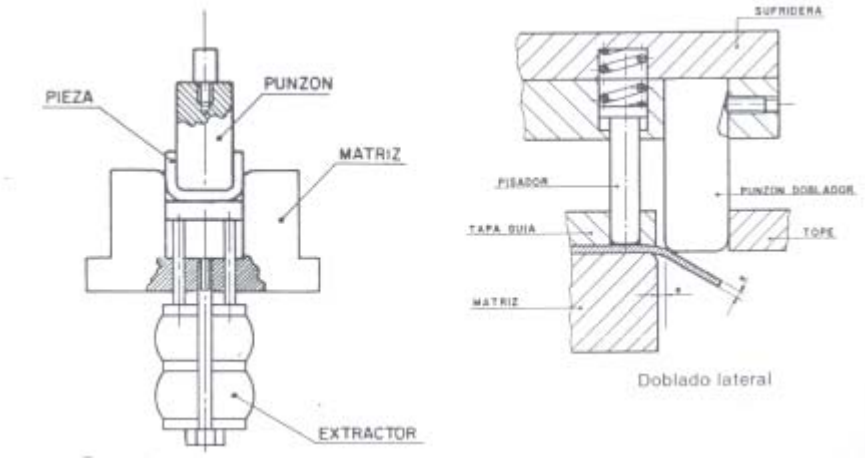


Figura 5. Troquel de doblado, segundo grupo.

El último grupo abarca a los troqueles más complejos, que efectúan embuticiones, doblados y punzonados. Figura 6.

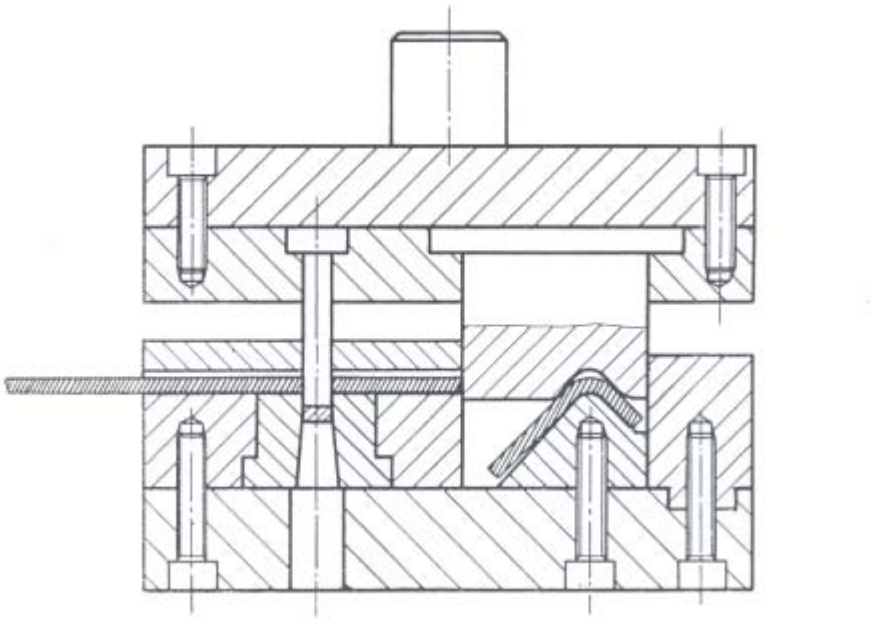


Figura 6. Troquel para punzonar, cortar y doblar con prensa inclinada.

Troquel de embutido

La técnica de embutido es la operación mecánica en el cual una chapa adquiere la forma de un recipiente al ser formado mediante punzón y matriz. La conformación definitiva de la pieza puede obtenerse en una o más etapas, dependiendo del grado de complejidad de la misma. En esta operación la lámina está sometida a esfuerzos de tensión y de compresión.
Con el objeto de que no se formen arrugas en las paredes de las piezas, se utiliza una plancha (prensachapas), lográndose además con está un mayor alargamiento de la chapa.

Los troqueles de embutir se clasifican en:

a) Troqueles de simple acción. En la superficie de la matriz se maquina un escalón para localizar o posicionar el disco o chapa de trabajo, la pieza embutida es empujada a través de la matriz y extraída del punzón mediante el escalón inferior de la matriz; el borde del recipiente se expande ligeramente haciendo posible la extracción. El dibujo de la figura 7 muestra dos tipos de troqueles de simple acción.

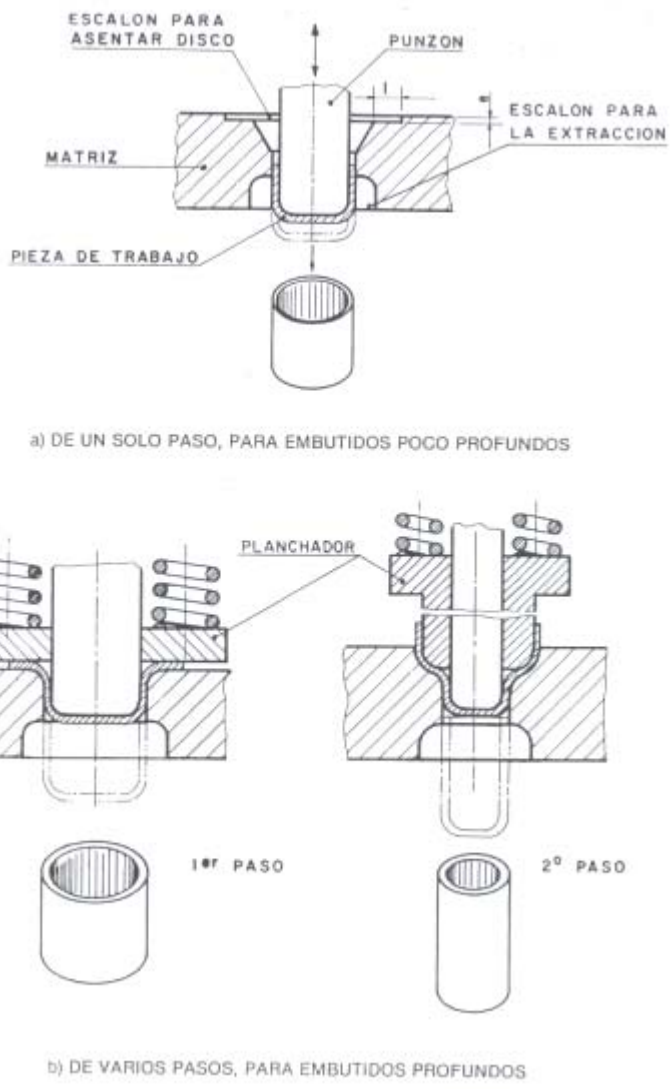
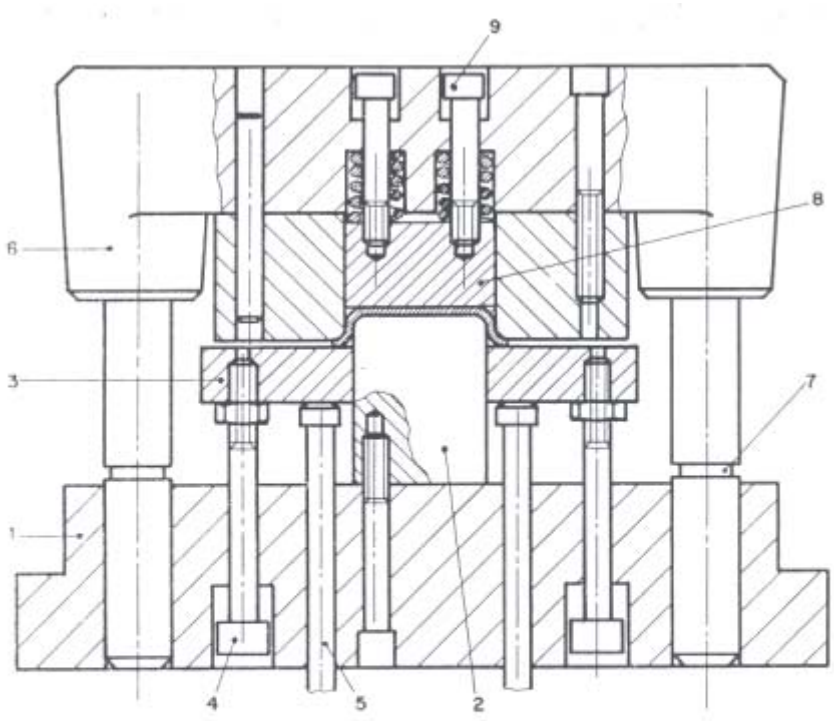


Figura 7. Troqueles de embutido simple acción.

b) Troqueles de doble efecto. Además de punzón y matriz cuentan con un extractor o almohadilla de presión, lo que permite efectuar embutidos más profundos y la formación de cejas y bordes. Ver figura 8.

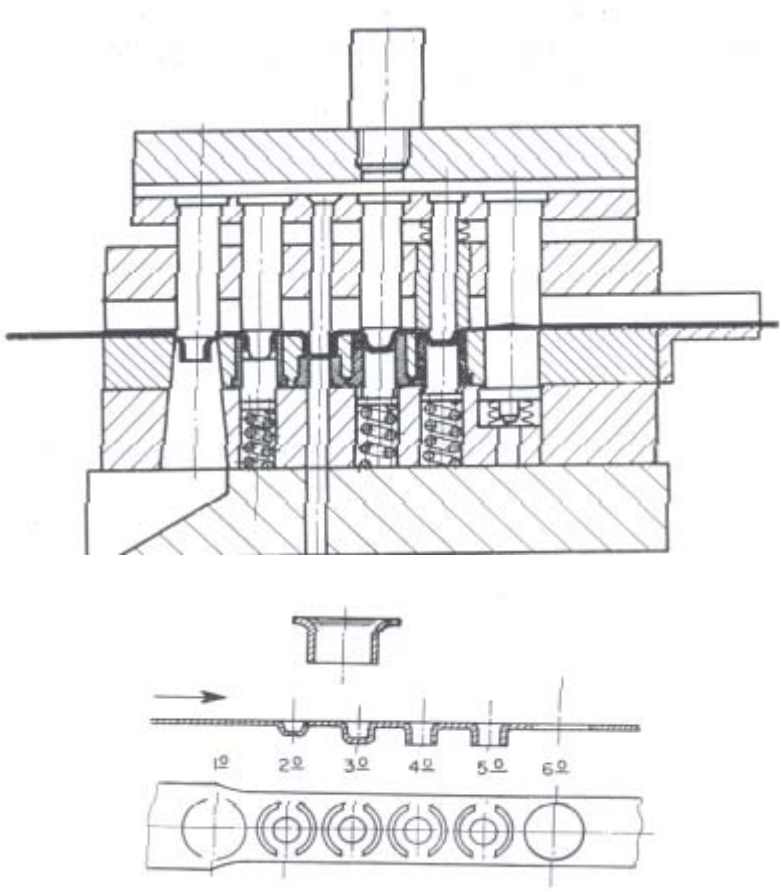


1.—ZAPATA INFERIOR
2.—PUNZON
3.—PLACA PRENSACHAPAS
4.—TORNILLO PARA LIMITAR LA CARRERA
5.—PERNO ACCIONADO POR COJIN NEUMATICO
6.—ZAPATA SUPERIOR
7.—COLUMNA GUIA
8.—PRENSACHAPAS Y EXTRACTOR
9.—TORNILLO PARA LIMITAR LA CARRERA DEL PRENSACHAPAS

Figura 8. Troquel de doble acción.

c) Troqueles progresivos. Se utiliza cuando la pieza debe permanecer unida al esqueleto de la tira progresiva hasta la última estación. Para este tipo de troquel se ilustra con la figura 9.

d) Troqueles progresivos con sistema de transferencia. Se utiliza cuando la pieza no necesita permanecer unida al esqueleto de una tira progresiva, lo que permite recortar la chapa o disco original e introducirlo al troquel de embutido.



1er PASO.- CORTE PARA MANTENER CONSTANTE EL ANCHO DE LA TIRA
2º PASO.-PRIMER EMBUTICION
3er PASO.-SEGUNDA EMBUTICION
4º PASO.-RECORTE DEL AGUJERO
5º PASO.-ESTIRAMIENTO Y CONICIDAD DE LA PARTE SUPERIOR
6º PASO.-SEPARACION POR CORTE

Figura 9. Troquel progresivo para embutir

Otras operaciones de corte son:

Ranurado. Se refiere al corte de agujeros alargados.

Perforado. La perforación designa el corte de un grupo de agujeros, por implicación pequeños y espaciados uniformemente en un patrón regular. El corte de muescas elimina material de un lado de la lámina o cinta.

Lanceado. Hace un corte parcial a través de una cinta.

Recortado o rebabeado. Es cortar el exceso de metal en una brida o la rebaba de una pieza.

Hendido. Corte del material original a lo largo pasándolo en forma continua a través de rodillos espaciados

Rasurado. Es un corte ligero que se aplica principalmente para mejorar la superficie de agujeros y eliminar orillas burdas.

Mordisqueo. Operación para cortar cualquier forma de metal en lámina sin herramientas especiales.

Cizallado. Corte en el que el punzón entra en el material, lo empuja hacia abajo dentro de una abertura en la matriz. Los esfuerzos en el material se vuelven más altos en los filos del punzón y de la matriz y el material comienza a agrietarse allí. La matriz es un poco más grande que el punzón por una cantidad de espacio llamada claro de ruptura, si este claro de ruptura es el correcto las grietas se encuentran unas con otras y la ruptura se completa. Ejemplos en la figura 10.

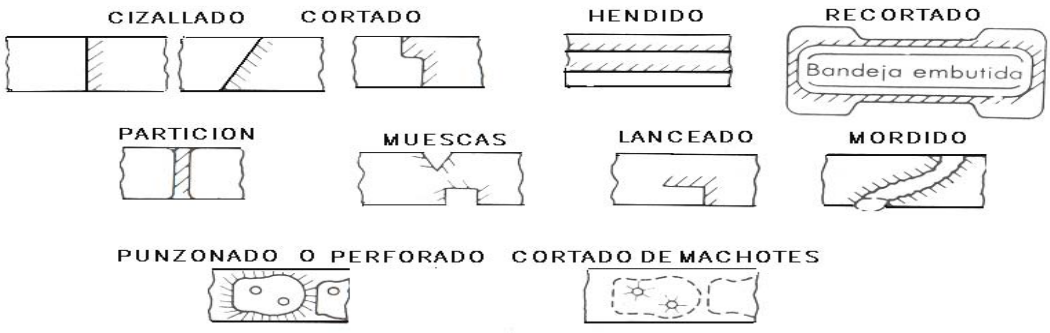


Figura 10. Diferentes tipos de corte.

Operaciones de doblado.

Las barras, varillas, alambres, tubos y perfiles estructurales lo mismo que la lámina de metal se doblan en muchas formas en dados o matriz.
Todo doblez de metal esta sujeto a esfuerzo más allá del límite elástico en tensión en el exterior y en compresión en el interior del doblez. El estiramiento del metal en la superficie exterior hace más delgado el material.
Algunas clases de dobleces en láminas de metal se muestran en la figura 11.

Extruído. Cuando el metal se comprime arriba de su límite elástico en una cámara y se le hace fluir a través y tomar la forma de una abertura.

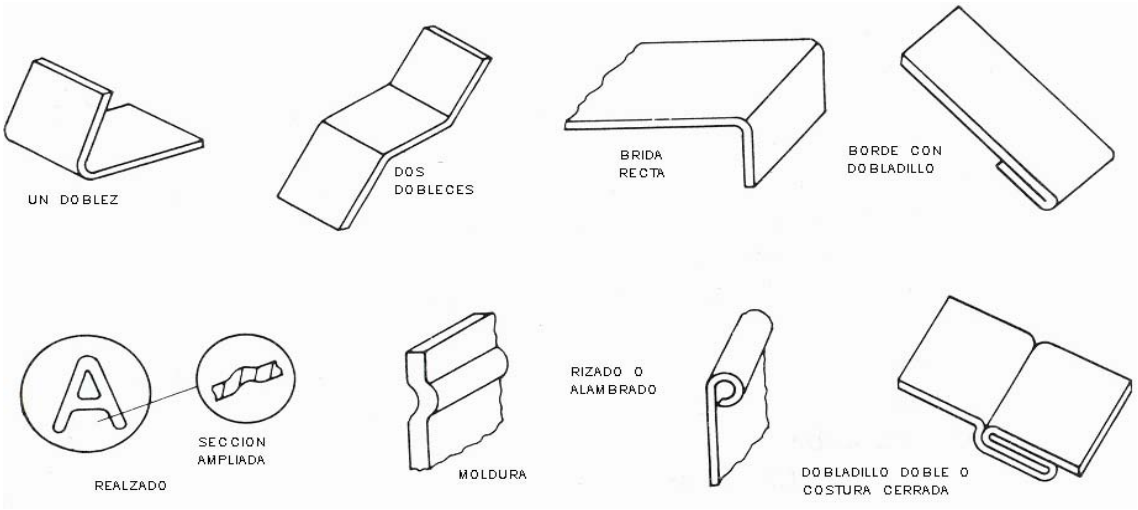


Figura 11. Diferentes operaciones de doblado.

Operación de embutido.

Las operaciones en esta categoría producen partes huecas de paredes delgadas o con forma de recipiente a partir de lámina de metal. Los ejemplos son recipientes sin costuras, bandejas, tinas, latas y cubiertas; paneles de automóviles, salpicaderas, techos y cofres, casquillos para cartuchos y granadas y reflectores parabólicos. La lámina de metal se estira cuando menos en una dirección pero con frecuencia se comprime también en otras direcciones en estas operaciones. La figura 12 muestra algunos embutidos.

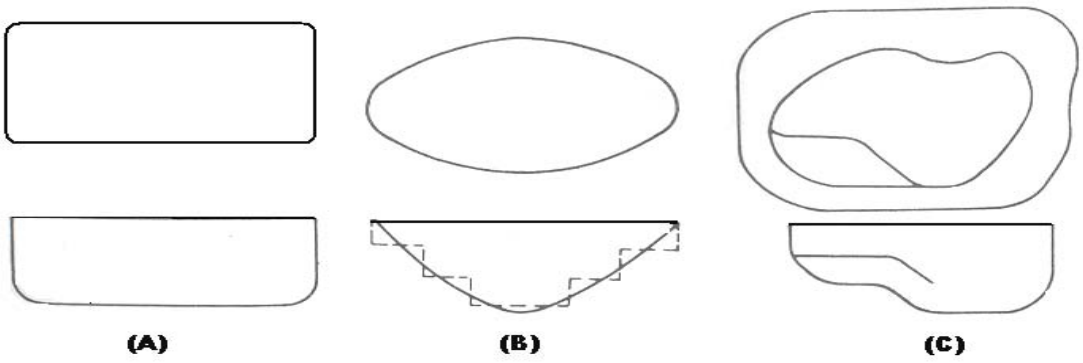


Figura 12 a) Embutido de piezas rectangulares, b) de piezas ovaladas y c) piezas complejas.

2.3 Elementos que constituyen un troquel.

Los elementos más representativos de un troquel se describen a continuación y en la figura 13 se señalan.

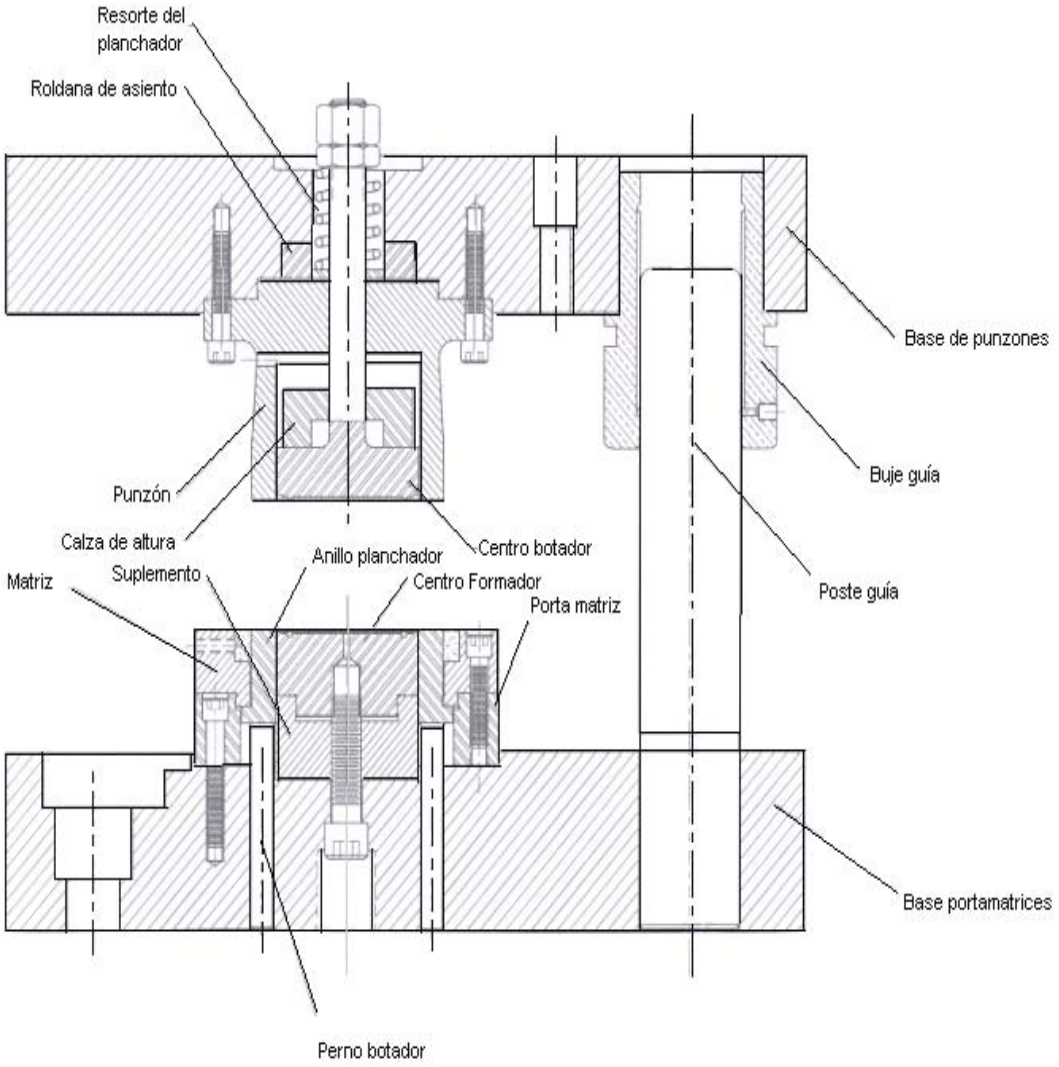


Figura 13. Partes más generales de los troqueles.

Punzón.
Son las partes móviles del troquel que se acoplan a la prensa mediante el porta punzones y tienen la figura total o parcial de la pieza que se quiere obtener.
Su movimiento debe ser estrictamente perpendicular a la matriz, por lo cual debe haber un ligero espacio entre sus paredes y los agujeros de la placa guía.
En el diseño de los punzones se debe evitar que sean demasiado débiles y se rompan.

Matriz.
Es pieza esencial del troquel, soporta grandes esfuerzos debido a los impactos.

Porta matriz.
Aloja y sujeta las matrices. Es indispensable por lo siguiente:

Comodidad de regular la alineación; facilidad de cambiar sólo la matriz.
La principal característica que debe tener un porta matrices, es poseer los medios adecuados para fijar la matriz y asegurar la base.

Base porta matrices. Tiene como función fijar la porta matriz a la mesa de la prensa, además de protegerla de flexiones durante la operación.
La base puede diseñarse para usarse solamente en un troquel o bien, normalizar el tamaño para que pueda usarse en troqueles distintos. El primer caso se presenta cuando su empleo es continuo, y las matrices son grandes; el segundo caso es recomendable para troqueles que permanecen largos intervalos de tiempo sin usarse y que tienen dimensiones pequeñas.

Placa guía. Tiene por objeto guiar a la tira progresiva durante su paso por el troquel, así como actuar de prensachapas o retensor de la chapa cortada, esto se debe a que el punzón tiende a levantar la tira debido a la entalladura del recorte sobre el punzón.
La entalladura se debe a la recuperación elástica del material después del troquelado.
La tapa guía tiene las mismas figuras que la placa matriz; a través de ellas pasan los punzones, por lo cual su fabricación debe ser tan cuidadosa como la misma placa matriz.

Perno botador. Es un sujetador con rosca externa que se inserta a través de orificios en las partes y se asegura con una tuerca en el lado opuesto.

Poste guía. Se encarga de mantener alineados el punzón y la matriz

Base de punzones.
Sostiene a los punzones, por medio de recalcado, dentro de una funda o casquillo o empotrado en un caja. La sujeción de punzones depende de los siguientes factores:

a) dimensiones del punzón
b) forma del punzón
c) técnica de fabricación del punzón
d) tipo de troquel
e) espesor de la chapa y tipo de material
f) cantidad de piezas a cortar.

2.4 Tipo de troqueles de la empresa.

La empresa utiliza troqueles que realizan las operaciones mencionadas anteriormente, se emplean principalmente para materiales metálicos como la hoja de lata o el aluminio en diversos calibres.

Los troqueles son principalmente de tres tipos sencillos, dobles y múltiples o progresivos, refiriéndose a una operación, dos operaciones o múltiples operaciones. Se utilizan sobre todo los dobles y progresivos para reducir etapas en la elaboración del producto.

Las máquinas con las que cuenta la empresa pueden ser automáticas o manuales lo que también determinan el tipo de troquel que se montara, pues las partes para su colocación son diferentes.

Algunos troqueles con los que trabaja la empresa se pueden ver en las siguientes imágenes (Figura 14)

Figura 14. Muestra de los troqueles del Grupo Zapata

Capítulo 3. Razonamiento Basado en Casos (RBC).

3.1 Introducción.

El Razonamiento Basado en Casos, permite resolver un problema mediante el empleo de problemas resueltos en el pasado similares al planteado. Da una nueva aproximación a la resolución de problemas y el aprendizaje incrementa su conocimiento almacenando el nuevo caso para ser usado en situaciones futuras.

Los casos o problemas resueltos en el pasado (también denominados ejemplos o casos de entrenamiento) se pueden representar por atributos que son las características del problema, hay un atributo especial que se denomina la clase.

Ventajas

El esfuerzo en la solución de problemas puede ser capturado para ahorrar trabajo en el futuro.
Experiencias previas que hayan sido exitosas pueden ser utilizadas para justificar nuevas soluciones.
Experiencias previas que no hayan sido exitosas se pueden utilizar para anticipar problemas.
La comunicación entre el sistema y los expertos se realiza en base a ejemplos concretos, es decir el sistema explica sus decisiones citando precedentes.
El RBC trabaja a partir de bases de datos existentes, no se requieren entrevistas con los expertos, simplificándose la adquisición del conocimiento.
El RBC es un algoritmo de aprendizaje incremental, el aprendizaje tiene lugar tan pronto como un nuevo ejemplo esta disponible, sin excesivo costo computacional.
El RBC permite proponer soluciones para los problemas rápidamente, evitando el tiempo necesario para derivar respuestas desde el estado inicial de un proceso de búsqueda de soluciones. Esta ventaja se manifiesta principalmente en situaciones donde un sistema basado en reglas, por ejemplo, hubiese requerido realizar una larga cadena de inferencias para alcanzar una solución.
El RBC permite proponer soluciones en dominios que no se comprenden completamente.
Los casos ayudan a focalizar el razonamiento sobre las partes importantes de un problema señalando que rasgos del problema son importantes.
El RBC es aplicable a un amplio rango de problemas.

Desventajas

1. Resulta difícil encontrar una estructura apropiada para describir el contenido de un caso y decidir cómo la memoria de casos debe ser organizada e indexada para un almacenamiento, recuperación y rehúso efectivos y eficientes.
2. Cuando el modelo del conocimiento general del dominio es incompleto, presenta información imprecisa, etc., resulta difícil la integración del mismo a la estructura de la base de casos.
3. Su principal limitación radica en la capacidad de representación de los valores, aspecto que dificulta su aplicación en situaciones que requieren representaciones de casos complejos como por ejemplo los que se presentan cuando existe un alto grado de interrelación entre las características. También hay que señalar que la definición de las funciones de semejanza y los mecanismos de selección de características para cada problema específico son problemas abiertos en este campo, cuya solución depende en gran medida de la experiencia de los expertos.

3.2 Elementos y su interacción.

Elementos del Sistema

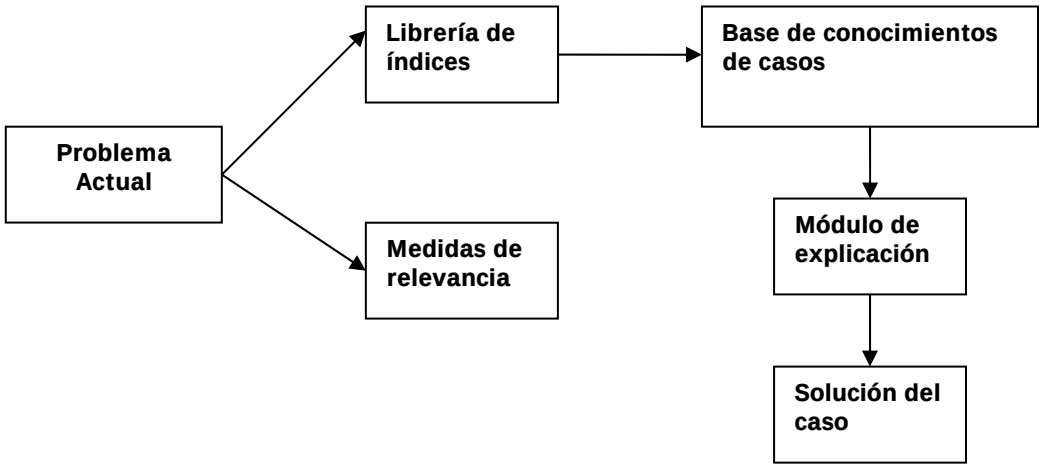


Figura 15. Elementos que componen a un Sistema RBC

Base de conocimiento de casos

Es una base de datos de casos históricos estructurada, que captura problemas reales y sus soluciones. Se diseña para almacenar conocimiento y experiencia en el problema.

Librería de índices

Formada por un conjunto de índices para buscar y recuperar casos similares al problema actual. El mecanismo de indexado determina los casos que serán seleccionados, mientras que el proceso de recuperación asegura que el caso más relevante es seleccionado para un análisis posterior.

Medidas de relevancia
Son un conjunto predeterminado de características del problema, generales o específicas, que el sistema usa para asegurar la relación entre el caso actual y los históricos. Con estas medidas el sistema puede seleccionar y clasificar los casos más relevantes.

Módulo de explicación

Es el que permite justificar y explicar el análisis completo del problema y las soluciones propuestas, así como la semejanza o diferencia entre dicha solución y las de los casos históricos.

El proceso completo que se realiza en un sistema de razonamiento basado en casos se puede representar como un ciclo de actividades:

1. Recuperar el o los casos más parecidos al problema actual. Para ello el sistema utiliza la librería de índices.

2. Reutilizar la información y el conocimiento de dicho caso para intentar resolver el problema.
3. Revisar la solución propuesta si es necesario.
4. Retener la parte útil de esta experiencia para ser utilizada en la resolución de futuros problemas.

Solución de un caso.

Un caso es una parte de conocimiento contextualizado representando una experiencia. Los casos tienen:

1. El problema que describe el estado del mundo cuando ocurrió el caso
2. Una descripción de la solución encontrada y/o
3. Un resultado describiendo el estado del mundo después de que ocurrió el caso

3.3 Metodología para un sistema RBC

El ciclo del Razonamiento Basado en Casos (Figura 16) está formado por los cuatro procesos siguientes:

1. Recuperar el caso o casos pasados más similares. Esto significa, retomar la experiencia de un problema anterior que se cree es similar al nuevo.
2. Reutilizar la información y conocimiento de este caso o casos recuperados para resolver el nuevo problema lo cual significa, copiar o integrar la solución del caso o casos recuperados.
3. Revisar la solución propuesta.
4. Guardar la nueva solución una vez que ha sido confirmada o validada. Se guardan aquellas partes de la experiencia de una manera tal que sea útil para resolver problemas futuros.

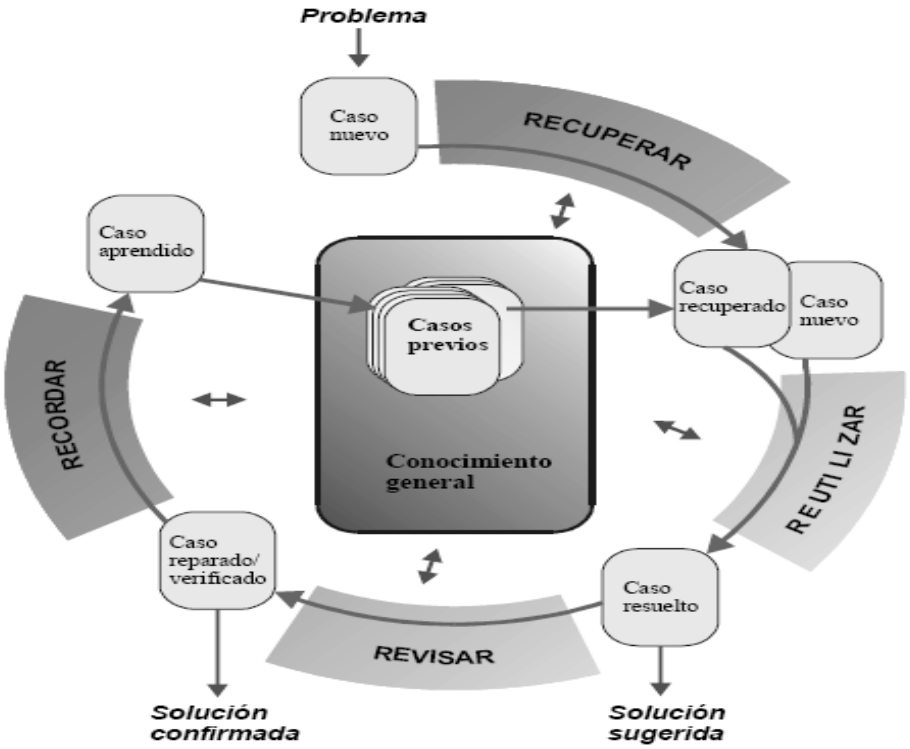


Figura 16. El ciclo del Razonamiento Basado en Casos y sus cuatro procesos.

El esquema siguiente ejemplifica más a detalle la manera en que trabaja la metodología de Razonamiento Basado en Casos.

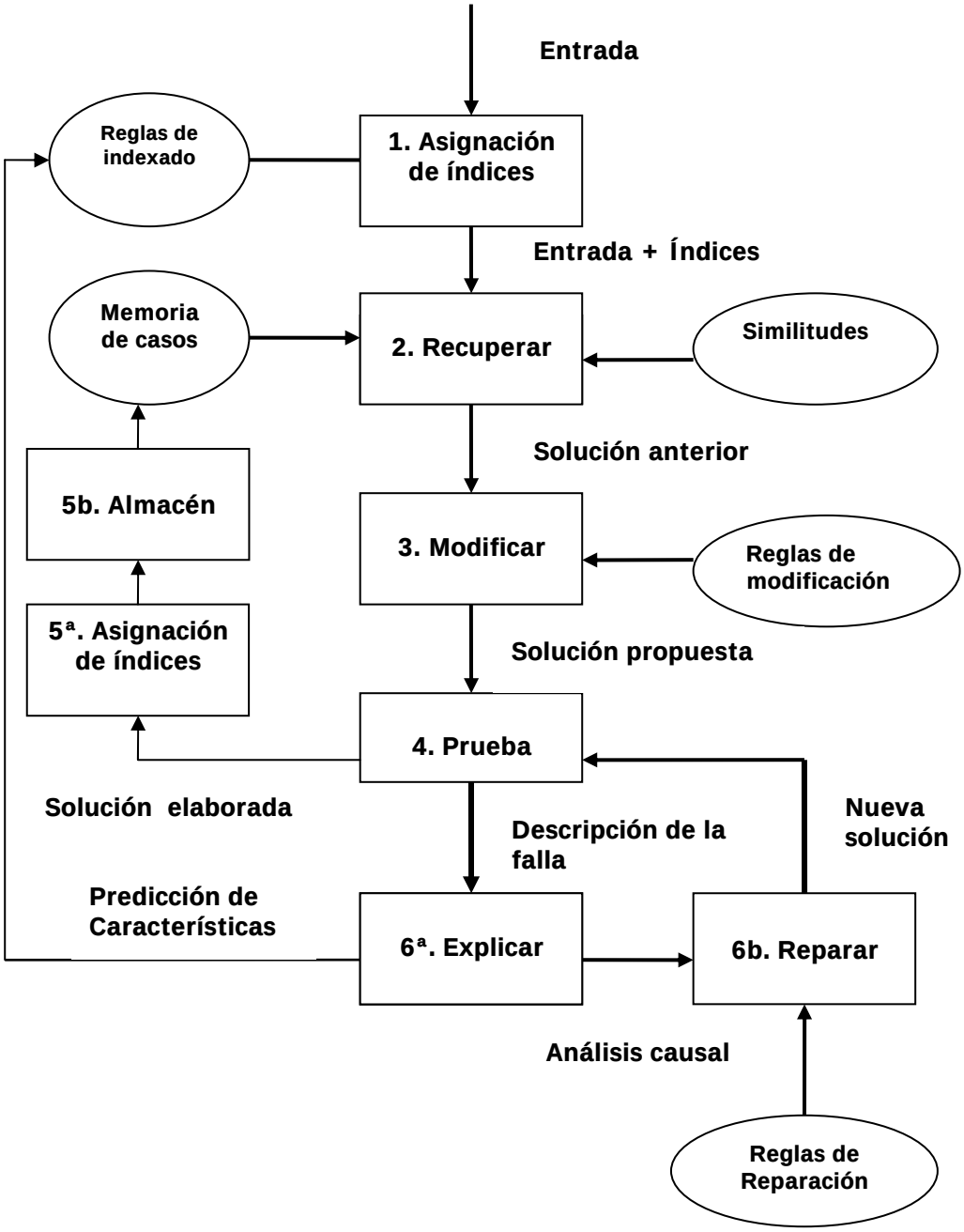


Figura 17. Ciclo del RBC.

1. Asignación de índices: A las características de los nuevos eventos son asignados índices.
2. Recuperar: Los índices son usados para recuperar un caso similar que se encuentra en memoria. El caso pasado contiene una solución anterior.

3. Modificar: La vieja solución es modificada conforme a la nueva situación, convirtiéndose en la propuesta de solución.
4. Prueba: La propuesta de solución es probada. Esta puede ser acertada o errónea.
5. Asignación y almacenamiento: Si la solución sucede, entonces se asignan índices y se almacena una solución elaborada. El plan satisfactorio es incorporado dentro de los casos de memoria.
6. Explicar, reparar y probar: Si la solución falla. Entonces se explica la falla, se repara la solución elaborada y se prueba de nuevo. El proceso explicativo identifica la fuente del problema. La predicción de rasgos del problema son incorporados dentro de las reglas de indexación para anticipar este problema en el futuro.

Reglas de indexación: identifican las características en la entrada que provee apropiadamente índices en el caso de memoria.

Existen tres formas de recuperar los casos, usadas conjuntamente:

1. Vecino más cercano: el sistema selecciona el caso almacenado en que más se parezcan sus características a las del problema en estudio. Las características pueden tener pesos para dar más importancia a unas que a otras.
2. Razonamiento inductivo: el sistema ordena los casos en memoria basándose en las características que más frecuentemente se parecen.
3. Indexado guiado por el conocimiento: el sistema selecciona las características más relevantes en función de la experiencia, implementada en forma de reglas.

Los índices deben de:

- ser predictivos
- contemplar el propósito para el cual el caso se va a utilizar
- ser suficientemente abstractos para ensanchar el uso de la base de casos
- ser suficientemente concretos para ser reconocidos en el futuro

Dentro de los métodos de indexación están:

- Indexación por atributos o dimensiones que son responsables de la solución o que influyen en el resultado (*checklist*)
- Indexación basada en diferencias: selecciona por índices que diferencian un caso de otro
- Métodos basados en similitudes o en explicaciones: produce un conjunto de índices para casos abstractos creados a partir de casos que comparten atributos comunes. Los atributos que no comparten se utilizan como índices de los casos originales.
- Métodos inductivos: identifica atributos predictivos y los utiliza como índices (variantes de ID3)
- Técnicas basadas en explicaciones: determina los atributos relevantes a cada caso e indexa por medio de ellos.

Memoria de caso: Es la memoria de episodios, la cual comprende la base de datos de experiencia.
Métrica de Similitud. La métrica de similitud puede ser usada para decidir cual caso es más parecido a la situación.

Reglas de Modificación. Ningún viejo caso va a ser exacto para una nueva situación. El viejo caso está modificado adecuadamente.
Reglas de reparación. Una vez identificados y explicados los errores se debe probar.

Tipos de sistemas RBC

El sistema puede pertenecer a alguno de los dos tipos básicos de sistemas RBC que existen, enseguida se mencionan las características que los definen y tomaremos un camino hacia la creación del mismo.

El tipo uno son los de interpretación de situaciones cuyo propósito es la clasificación o formulación de un juicio. Estos sistemas son principalmente los que hacen diagnósticos médicos o de fallos en equipos de cómputo.

El otro tipo de sistemas RBC son los que se dedican a la resolución de problemas y su objetivo principal es aplicar la solución de un problema pasado para obtener la solución de un problema actual.

Su empleo es principalmente en el diseño y la planificación.

En interpretación de situaciones se realiza el siguiente proceso:

- Determinar las características relevantes de la situación actual.
- Recuperar casos ya clasificados o interpretados con características similares.
- Comparar la situación actual con los casos recuperados para determinar qué interpretación es aplicable. Utilizando los casos previos como justificación de la elección.
- Guardar la situación actual junto con su interpretación y su justificación como un nuevo caso.

En resolución de problemas el proceso es así:
- Determinar las características relevantes del problema.
- Recuperar casos que resuelven problemas similares.
- Adaptar la solución de los casos recuperados al problema actual.
- Guardar el problema actual junto con su solución como un nuevo caso.

La necesidad que se pretende cubrir es que en el momento de recibir la solicitud de un nuevo producto (Problema) se encuentre un producto ya hecho (Problema anterior) igual o parecido cuyos troqueles (Solución) se puedan utilizar para fabricarlo.
Con apoyo de lo anterior, el camino a seguir es el tipo de sistema enfocado a la "resolución de problemas" ya que nuestra necesidad es aplicar las soluciones de problemas anteriores a nuevos problemas.

Siguiendo el proceso para los sistemas de este tipo se determinan las características relevantes del problema, que desde ahora en adelante se llamará “Caso”.
Entonces las características del caso serán:

1. Familia a la que pertenece el producto.
2. Número de elementos que lo componen.
3. Geometría
4. Ancho
5. Largo
6. Profundidad
7. Diámetro
8. Capacidad
9. Tipo de lámina
10. Calibre
11. Tipo de producto

Caso	Componente	Familia	CR	Geometría	A	L	P	D	C	Lámina	Calib	TipoDeProducto
1	Fondo	Alcoholero	3	Cuadrado	238	238	337	0	18gm	ETP	90	Con perforación
2	Fondo 12 grs	Cajas y varios	1	Redondo	0	0	0	0	10gm	Aluminio	40	Estándar
3	Cubierta 40 grs	Cajas y varios	1	Redondo	0	0	0	0	40gm	Aluminio	40	Estándar
4	20mm	Casquillos	0	Redondo	0	0	13.6	20	0	Aluminio	50	Farmacéutico
5	30x22 mm	Casquillos	0	Redondo	0	0	22	30	0	Aluminio	50	Bebidas
6	414x108	Ceras	0	Redondo	0	0	0	0	3gm	Aluminio	50	Estándar
7	Fondo 306x303x010	Cigatam	2	Rectangular	306	303	10	0	0	TFS	54	Estándar
8	Cono 202	Conos y Fondos	1	Redondo	0	0	98	52	125gm	ETP	65	Pared recta
9	Fondo 211	Conos y Fondos	1	Redondo	0	0	102	65	250gm	TFS	70	Estándar
10	202	Conos y Fondos	1	Redondo	0	0	0	52	177.4gr	ETP	85	Acordonado
11	Long Jhon	Filtro	2	Redondo	0	0	0	11	0	ETP	85	Long Jhon
12	202	Galletero (Cuerp	1	Redondo	0	0	30.5	52	0	ETP	85	Cuerpo redondo
13	914	Galletero (Cuerp	1	Redondo	0	0	5.08	250	0	ETP	78	Cuerpo redondo
14	507/603x611	Galletero (Fondo	1	Cónico	136	169	169	0	0	ETP	85	Estándar
15	410x410	Galletero (Fondo	1	Octagonal	111	111	127	0	0	TFS	78	Estándar
16	Fondo 710	Galletero (Fondo	1	Redondo	0	0	0	184	0	ETP	85	Estándar
17	TSP 310x310	Galletero (Tapas	1	Cuadrado	92	92	10.2	0	0	ETP	85	Estándar
18	Cubierta 310x210x005	Stanford	2	Rectangular	310	210	5	0	0	ETP	85	Estándar
19	20-400	Tapas de rosca	0	Redondo	0	0	0	22	0	Aluminio	45	Plana
20	C-1-450	Tapas de rosca	0	Redondo	0	0	0	55	0	Aluminio	45	C/Canaleta

Tabla 1 Base de caso

CR.- Componentes relacionados
A.- Ancho [mm]
L.- Largo [mm]
P.- Profundidad o altura [mm]
D.- Diámetro [mm]
C.- Capacidad
Cal.- Calibre de lámina

La base de conocimiento (Tabla 1) está compuesta por 20 ejemplos, donde el campo correspondiente a Producto es el nombre del caso, y cada atributo tiene un valor que es de tipo numérico o de tipo cadena.

Las razones por las que el tipo de valor de los atributos es un dato numérico o cadena de caracteres, son que el número de atributos no es muy grande lo que permite hacer comparaciones directamente entre los valores de cada caso.

Recuperación

Las características o atributos de cada caso tienen asociado un valor, este valor es el índice que permitirá hacer la recuperación de los casos con más similitudes, para esto se necesita aplicar una técnica que por medio de los índices dados proporcione un valor que represente la similitud con el caso a resolver.

Debe tomarse en cuenta que algunos atributos toman valores dentro de un cierto umbral.
El procedimiento de búsqueda dependerá de la organización (la estructura de datos) de los casos.

La ordenación (ranking) de los casos recuperados debe estar basada en el porcentaje de similitud, es decir el primero en la lista es el caso cuyo porcentaje se acerque más al 100 por ciento.

Selección del caso mejor. La selección del mejor caso estará basada en la solución que mejor ayude a resolver nuestro problema.

Indexación

En RBC los índices sirven para localizar los casos relevantes

El vecino más cercano

El método del vecino más cercano no es más que obtener la distancia que hay entre los atributos del caso en la base y el caso nuevo.

Para medir la similitud total entre dos casos, se podía utilizar la similitud euclidiana o la de Hamming.

Similitud Euclidiana:

$$Sim(X,Y) = 1 - \sqrt{\Sigma(wi^2 * dist(xi, yi)^2)}$$

Similitud de Hamming.

$$Sim(X,Y) = 1 - \frac{\sum_{i=1}^{n} wi * dist(xi, yi)}{\sum_{i=1}^{n} wi}$$

Donde X es un caso, x es el valor de un atributo de un caso.
Y es otro caso, y es el valor de un atributo de otro caso.
dist es la distancia entre atributos.
Sim es la similitud entre dos casos.
w es la ponderación de cada atributo.

Estableciendo los criterios para determinar la distancia entre valores de atributos, podemos decir que

Si ***x*** igual a ***y*** la distancia es 0
Si ***x*** diferente a ***y*** la distancia es 1
Si ***x*** es un valor numérico y en valor se acerca en ± 5 unidades a ***y*** la distancia es 0.5
Para esto tenemos que, sí el valor de **x** es 43 y **y** es 43 la distancia es 0
Ahora si **x** = 43 $\pm$ 5 se acerca a **y** la distancia es 0.5
Cuando **y** no esta entre el rango de **x** $\pm$ 5 y desde luego no es igual a **x** la distancia se considera 1

Los pesos para cada atributo son

Atributo	Peso
Familia a la que pertenece el producto.	5
Número de elementos que lo componen.	1
Geometría	2
Ancho	2
Largo	2
Profundidad	2
Diámetro	2
Capacidad	2
Tipo de lámina	1
Calibre	1
Tipo de producto	1
Total de operaciones en su proceso	1

Tabla 2 Ponderaciones para cada atributo

3.4 Aplicaciones.

Estas son algunos de los campos donde se han realizado aplicaciones del Razonamiento Basado en Casos.

Las principales aplicaciones en que se emplea es en sistemas que requieran adquisición de conocimiento, como es el tema en el que se trabaja.

Áreas como la de leyes también hace uso de esta metodología, ya que se manejan casos, que se pueden repetir o que pueden parecerse a otros anteriores.

La medicina lo emplea en situaciones donde no hay explicación en la muerte, o anomalías en hombres y animales. Se puede prevenir nuevas enfermedades y realizar nuevos diagnósticos médicos.

Otras aplicaciones se dan dentro del diseño mecánico y arquitectónico, en la planeación de manufactura, reparación y adaptación.

La cocina no descarta el RBC para nuevas recetas, y los deportes tampoco.

Las aplicaciones comerciales y financieras son las principales en hacer uso de éste y otros métodos que componen los sistemas expertos, su empleo facilita el manejo de la información.

Estos son algunos ejemplos de las aplicaciones del RBC, y se puede decir que su uso es aplicable en la vida cotidiana

Capítulo 4. Análisis del sistema.

Modelo de ciclo de vida.

El ciclo en espiral es adecuado para el proyecto porque contempla los posibles cambios que se puedan suscitar durante el desarrollo del sistema, contempla los riesgos y permite hacer correcciones, se puede también introducir nuevas herramientas de desarrollo. Figura 18.

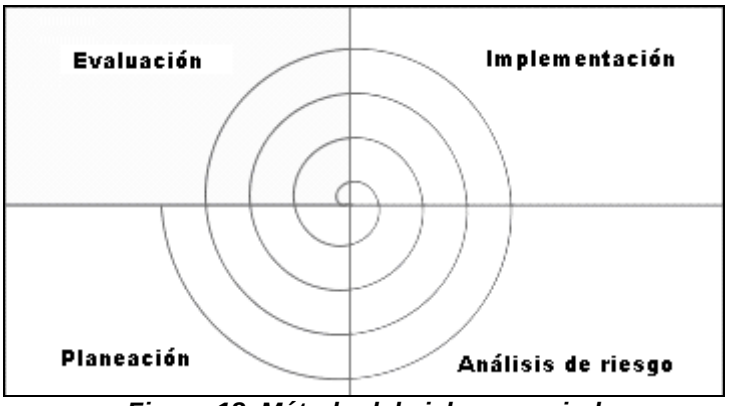


Figura 18. Método del ciclo en espiral

Partes del método.

- **Planeación.**

Análisis de los requerimientos

- **Análisis de riesgo**

Depuración de errores en los registros de información.

- **Implementación**

Diseño abstracto
Diseño concreto

- **Evaluación**

Validación y verificación del sistema.

4.1 Requerimientos

1. Debe contar con una interfaz de usuario totalmente amigable
2. El sistema debe permitir registrar los productos (componentes), por las siguientes características:

- Medida nominal
- Familia a la que pertenece
- Componentes con los que se relaciona
- Número de operaciones
- Relación con otros productos
- Dimensiones (Ancho, Largo, Profundidad y Diámetro)
- Capacidad
- Tipo de lámina
- Calibre
- Tipo de producto

3. El sistema debe permitir el registro, baja y cambios de la siguiente información de los troqueles.

- Familia para la que se emplea el troquel
- El número de operaciones que se hace dentro de cada familia
- La operación que realiza el troquel dentro de las operaciones de la familia
- Tipo de operación
- Componente en el que se emplea el troquel
- Frecuencia en su uso
- Ubicación física en el momento del registro
- Localización
- Tipo de troquel
- Observaciones

4. Cada troquel debe tener relacionadas las partes que lo componen teniendo en cuenta el tipo de operación que realiza.

5. Se necesitan hacer consultas sobre esta información, eligiendo los diferentes campos que definen al troquel y se deben visualizar los resultados.

6. Para la parte del sistema en la que se buscaran troqueles que se puedan reutilizar deben tomarse en cuenta las características que se nombraron para el registro de los productos.

7. Los resultados de la búsqueda debe proporcionar información acerca del producto y del troquel, tomando en cuenta las características que se establecieron para el registro de los troqueles.

8. Debe de contar con restricciones de usuarios, así como un módulo donde se pueda llevar el registro de los troqueles que salgan del almacén.

9. La aplicación está destinada al área de almacén donde solamente habrá un equipo de cómputo en la que será instalado.

4.2 Planeación

Análisis de requerimientos

1. Interfaz de usuario

Para la creación de un ambiente amigable se elige un software de programación que permita crear una aplicación en la cual el usuario encuentre una apariencia familiar y tener la facilidad de uso.

2. Registro de productos

El registro de productos, que son los componentes de los envases que se elaboran, los describimos por estas características:

Medida nominal. Es la medida con la que identifican el diseño de un componente, esto facilita el hecho de que puede haber infinidad de marcas que manejan éste mismo

diseño y registrar varias veces el mismo componente, sin embargo como lo que se busca son el o los troqueles con el que se elabora este producto, es mejor referirse a sus medidas nominales.

Familia a la que pertenece. La familia es el área de producción en donde se fabrica cada producto. Las familias son las siguientes (figura 19):

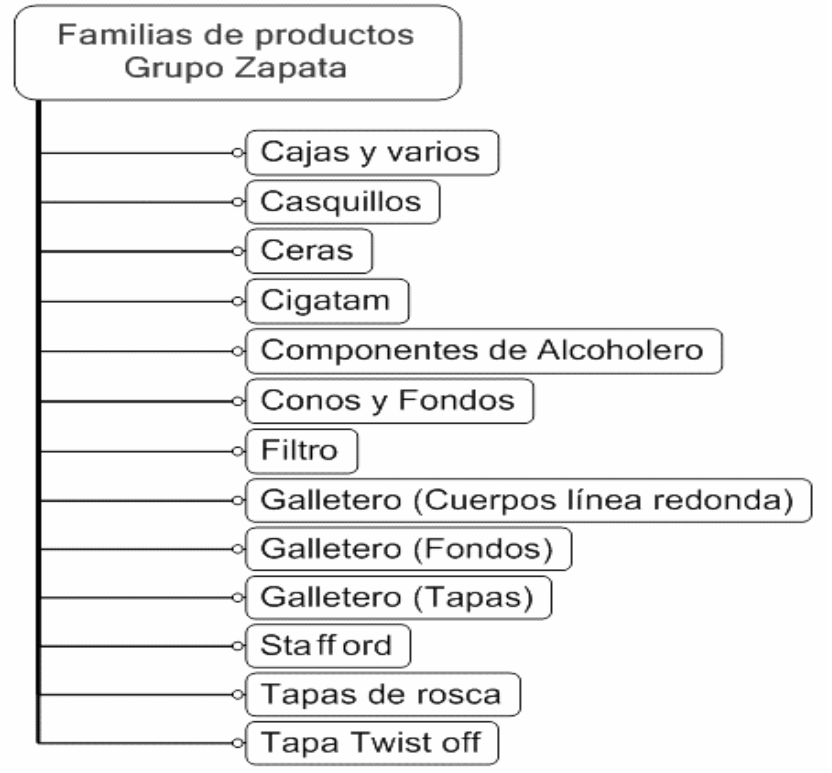


Figura 19. Familias o áreas de producción.

- Componentes que lo integran
  Cada producto esta integrado por varios componentes, por ejemplo.

Cubiertas
Tapas
Fondos
Ensambles
Boquillas
Asas
Plato soporte
Anillo reten
Cuerpo
Cedazo

Figura 20. Componentes de los productos

- Dimensiones (Ancho, Largo, Profundidad y Diámetro)
  Las medidas del producto generalmente se dan por estas dimensiones y en algunos casos si no son conocidas se registra por la capacidad que tienen.
- Tipo de lámina
  La empresa maneja tres tipos de láminas: Aluminio, ETP y TFS.

- Calibre. Se manejan diferentes calibres según el tipo de lámina.
- Tipo de producto
  Puede haber varios productos con medida nominal igual sólo que lo distingue de otro alguna característica extra.

3. Registro, baja y modificación de la información de los troqueles.

Para iniciar el registro de un troquel necesitaremos definir como estará compuesto el código que lo identificará, las características más representativas del troquel nos servirán para su composición. Los dígitos del código son una combinación de letras del alfabeto y números enteros del 0 al 9.

El primer dígito nos lo dará la "Familia", por lo que para cada una de las familias asignaremos un carácter que la represente (ver Tabla 3). Un criterio para asignarle su carácter es la letra con la que inicia su nombre y si varias inician con la misma letra tomamos otra más que no se use anteriormente.

Identificador	Nombre de la Familia
C	Cajas y Varios
Q	Casquillos
E	Ceras
M	Cigatam
A	Componentes Alcoholero
N	Conos y fondos
F	Filtro
G	Galletero (Cuerpos línea redonda)
D	Galletero (Fondos)
T	Galletero (Tapas)
S	Stafford
W	Tapa Twist off
R	Tapas de rosca

Tabla 3. Carácter que identifica a la familia

Cada familia tiene un cierto número de operaciones por las que pasan sus productos y en la que se puede utilizar un troquel. De aquí establecemos que el segundo dígito indica el número total de operaciones por familia, un tercer dígito indicará, dentro de ese total, que número de operación hace y el tipo de operación (tabla 4), que sería representado por el cuarto dígito.

Geometría del componente que realiza el troquel es el dígito quinto, en la tabla 5 se ve el carácter que identificará a las formas geométricas más comunes en los productos.

Identificador	Operación
1	Corte
2	Costillado
3	Engomadora de Plastisol.
4	Ensambladora de Linner.
5	Estirado
6	Expansión
7	Formado de Bisagra
8	Formado de muescas
9	Formado de rizo y anclas
A	Iniciado
B	Perforado
C	Perforar borde
D	Reborde
E	Recortar pestaña
F	Recorte
G	Rizo y Cordón
H	Roscado
J	Troquelado
K	Troquelar lámina
L	Troquelar tira

Tabla 4. Identificadores de los tipos de operación

Identificador	Geometría
1	Triangulo
2	Cuadrado
3	Rectángulo
4	Redondo
5	Cilindro
6	Cono
7	Hexágono
8	Octágono
9	Cubo
A	Decágono
B	Elipse
C	Esfera
D	Parábola
E	Pentágono
F	Rombo
G	Romboide
H	Semiesfera
J	Trapecio

Tabla 5. Identificadores de las geometrías.

El sexto dígito en nuestro código representara las variables dimensionales, que son las que definen el tamaño del componente. Tabla 6

Identificador	Dimensiones
1	Ancho
2	Largo
3	Profundidad
4	Diámetro
5	Ancho y Largo
6	Ancho y Profundidad
7	Largo y Profundidad
8	Diámetro y Profundidad
9	Ancho, Largo y Profundidad

Tabla 6. Identificadores de las variables dimensionales

El dígito identificador en la posición 7 indica el tipo de troquel que se representa con la letra A si es automático y con la letra M si es manual.

Los dígitos 8 y 9 del código señalan cuantos troqueles con características iguales hay, es decir puede haber de 01 a 99 troqueles que tienen los mismos dígitos de la posición 1 a la 7 del código y estos dos dígitos serían los diferenciadores.

El código queda representado de la siguiente manera:

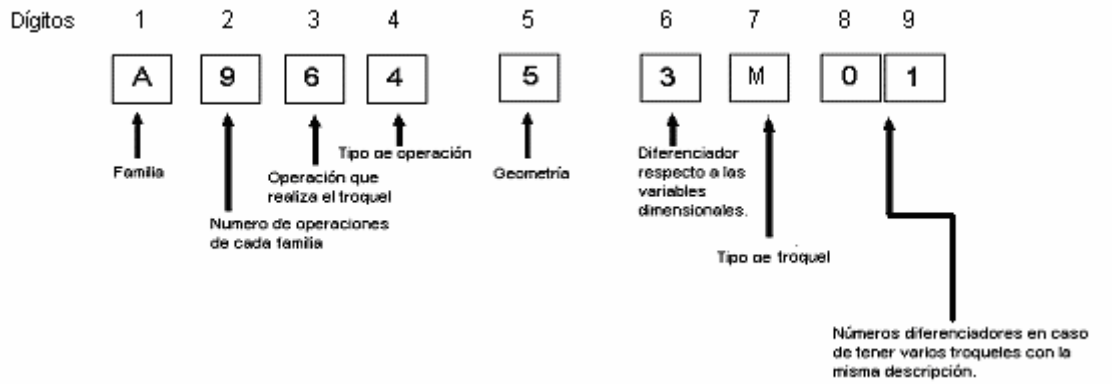


Figura 21. Representación del código de un troquel

Continuando con el punto 3 de los requerimientos, tenemos que se deben contemplar estos puntos en el registro del troquel:

- Familia para la que se emplea el troquel
- El número de operaciones que se hace dentro de cada familia
- La operación que realiza el troquel dentro de las operaciones de la familia
- Tipo de operación
- Componente en el que se emplea el troquel
- Frecuencia en su uso
- Ubicación física en el momento del registro
- Localización
- Tipo de troquel
- Observaciones

Algunos de ellos son parte necesaria para crear el código, estos serán los campos que no se podrán modificar de los registros por que su código ya no estaría representando las características del troquel.

4. Cada troquel debe tener relacionadas las partes que lo componen teniendo en cuenta el tipo de operación que realiza.

Los troqueles dependiendo de la operación que realizan manejan diferentes piezas, al registrar un troquel es indispensable marcar también sus partes que deben ser mostradas al elegir su operación.
Las características de las partes son el material con el cual están hechas y la cantidad de piezas.

5. Se necesitan hacer consultas sobre esta información, eligiendo los diferentes campos que definen al troquel y se deben visualizar los resultados.

Las consultas principalmente serán sobre la información de los troqueles asociada con sus partes y el producto para el cual se emplean.

6. Para la parte del sistema en la que se buscaran troqueles que se puedan reutilizar deben tomarse en cuenta las características que se nombraron para el registro de los productos.

Los elementos de búsqueda son estos:

- Medida nominal
- Familia a la que pertenece
- Componentes con los que se relaciona
- Número de operaciones
- Relación con otros productos
- Dimensiones (Ancho, Largo, Profundidad y Diámetro)
- Capacidad
- Tipo de lámina
- Calibre
- Tipo de producto

7. Los resultados de la búsqueda deben proporcionar información acerca del producto y del troquel, tomando en cuenta las características que se establecieron para el registro de los troqueles.

Se muestran los registros de la base de datos relacionados con los troqueles que devuelva la búsqueda

8. Debe de contar con restricciones de usuarios, así como un módulo donde se pueda llevar el registro de los troqueles que salgan del almacén.

Las restricciones de acceso son importantes para mantener en orden el contenido de la base de datos y el buen funcionamiento del sistema.

4.3 Recursos de software y hardware

El proyecto se crea bajo el sistema operativo Windows Xp edición Profesional, y con Visual Basic.net como plataforma de desarrollo.

La elección de este lenguaje de programación se debe principalmente a las características que debe tener la interfaz de usuario, ya que nos permite diseñar un ambiente amigable en el que los usuarios se encontrarán familiarizados con el aspecto tradicional de las ventanas de Windows que es el sistema operativo con el que tienen más contacto.

La posibilidad de que el sistema crezca en cierto momento es motivo también para emplearlo, da la facilidad de que las aplicaciones sean usadas en Internet.

Este software de desarrollo nos proporciona también la factibilidad de conectarnos con bases de datos, que en nuestro caso será una base local.
Para la base de datos empleamos Access, las razones para elegirlo son en primer lugar que los registros no sobrepasarán su capacidad y la segunda es que la aplicación en un principio esta contemplada para ubicarse únicamente en la máquina del almacén donde estarán localizados los troqueles.

Dentro del diseño se maneja Argo que es una aplicación para los esquemas UML y también DBDesigner que nos ayuda en el diseño de la base de datos.

El hardware disponible para programar las aplicaciones es una computadora con procesador Pentium 4 a 2.66 Ghz y 512 MB en RAM.

Capítulo 5. Diseño del Sistema.

5.1 Diseño de la base de datos

A continuación se establecen las entidades y los atributos de la base de datos.
La primera tabla contiene los atributos que definen generalmente un producto enfocado principalmente a su manufactura, los atributos son los más generales, por lo cual será utilizada como base de conocimiento para nuestro algoritmo de razonamiento basado en casos.

Tabla: BaseDeCasos	
Atributos	Descripción
Producto	Medida nominal con la que se conoce
Familia	Área en la que es producida
NoElementos	Componentes relacionados
Geometria	Geometría del componente
Ancho	Dimensión de longitud
Largo	Dimensión de longitud
Profundidad	Dimensión de longitud
Diametro	Dimensión de longitud
Capacidad	Cantidad de masa que puede contener
TipoDeLamina	Material que se utiliza para su fabricación
Calibre	Grosor de la lámina
TipoDeProducto	Diferentes formas que puede tomar un producto
TotalOp	Número de pasos para terminar el producto

La tabla Familia contiene los registros de las áreas de producción y a cada una se le asigna una letra que indique de quien se trata, y ya que la familia es la que define un producto es el dígito inicial del código que identifica al troquel.

Tabla: Familia	
Atributos	Descripción
IdFamilia	Identifica a la familia con un carácter
Familia	Nombre de cada área de producción

Todos los productos son en un principio identificados por la forma geométrica en esta tabla estarán las geometrías más comunes y tendrán un número o letra que los identifique también por que son parte del código del troquel

Tabla: Geometria	
Atributos	Descripción
idGeometria	Carácter con el que se identifica la geometría
Geometria	Forma que adopta un producto

La solución de cada problema se encontrara almacenada en esta tabla ya que estará relacionada con la tabla “BaseDeCasos”, cada producto almacenado tiene relación con varios troqueles que son utilizados para fabricarlo.

Tabla: Troquel	
Atributos	Descripción
Codigo	Caracteres que identifican un troquel
Familia	Área en la que se emplea el troquel
OpDeFamilia	Total de etapas para elaborar el producto
Operación	Número de operación en la que es empleado
TipoDeOperacion	Operación que realiza para crear el producto
Producto	Producto en el que participa este troquel
Geometria	Geometría que le da al producto
Ubicación	Lugar donde se encuentra
Localizacion	Número o espacio dentro del lugar donde está
Uso	Frecuencia con la que se ocupa
Observaciones	Comentarios sobre su estado físico
Imagen	Nombre de la imagen correspondiente al troquel

Cada troquel está compuesto por varias partes, se define el tipo de material, cantidad y el estado para contar con troqueles completos.

Tabla: Partes	
Atributos	Descripción
Codigo	Identificador del troquel al que pertenece la parte
Parte	Nombre de la parte del troquel
Material	Tipo de material con la que esta echa la parte
Cantidad	Número de partes del mismo tipo
EstadoFisico	Aspecto de las piezas

Las operaciones son técnicas de manufactura que se utilizan para los procesos de fabricación de los productos, cada troquel realiza una de estas técnicas y es importante identificar fácilmente la operación a la que pertenece, motivo por el cual su identificador es un carácter que tendrá uso en la creación del código del troquel.

Tabla: TipoDeOperacion	
Atributo	Descripción
idOperación	Identifica el tipo de operación
TipoDeOperacion	Nombre de la operación

Cada troquel de acuerdo a la operación que realiza tiene partes básicas y otras que no todos tienen ya que son las que definen la técnica, es por eso que en esta tabla se contendrán las partes de un troquel por operación relacionada con la operación a la que pertenece.

Tabla: PartesxOp	
Atributos	Descripción
idOperacion	Identifica el tipo de operación
Parte	Parte de un troquel respecto al tipo de operación

La tabla contiene los componentes que se relacionan con el producto registrado ya que en conjunto se obtiene un envase completo.

Tabla: RelComponentes	
Atributos	Descripción
Producto	Medida nominal con la que se conoce
Componente	Componente relacionado con el producto

Esta tabla tiene el propósito de tener almacenados los usuarios que harán uso de las aplicaciones.

Tabla: Usuarios	
Atributo	Descripción
idUsuario	Nombre de usuario
Nombre	Nombre del usuario que emplea el sistema
Contraseña	Contraseña de acceso al sistema
Privilegio	Nivel de funciones a las que tiene acceso

Para tener conocimiento de la salida de los troqueles es importante registrarlos, en esta tabla se contendrán para tener mejor control.

Tabla: Prestamo	
Atributo	Descripción
idPrestamo	Número de préstamo
Codigo	Código del troquel prestado
Solicitante	Nombre de la persona que lo pide
FechaSalida	Fecha del préstamo
FechaEntrega	Fecha en la que lo devuelven

El número de tablas de la base de datos es 10

5.2 Diseño de la interfaz

Para el diseño de la interfaz se emplea el modelo de Casos de uso que son diagramas de UML (Lenguaje Unificado de Modelado), con esto se puede observar la interacción de los usuarios con el sistema. Figura 22.

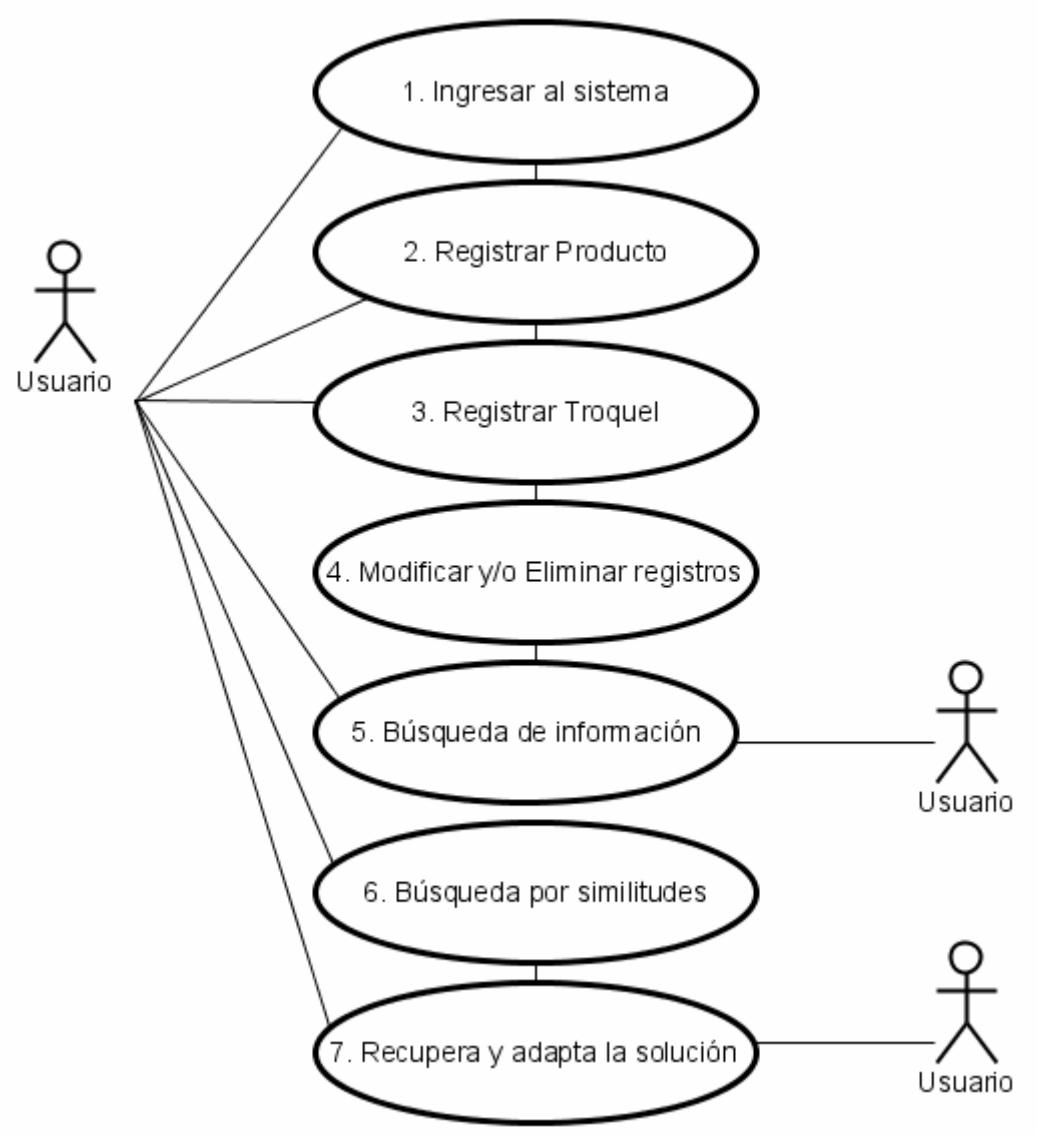


Figura 22. Diagrama de casos de uso.

Descripción de los Casos de Uso.

En esta parte se desarrolla más a fondo las actividades que se realizaran en cada Caso de Uso, se lleva una secuencia de pasos en los que puede estar involucrado el usuario, el sistema o presentarse un evento alterno que sea necesario para el desarrollo del sistema o que informe directamente al usuario de una situación de confirmación, de error o riesgo.

Cada tabla tiene el nombre del caso de uso y el número que lo identifica en el diagrama.

Caso de uso: Ingresar al sistema IDCU: 1			
Paso	Usuario	Sistema	Alterno
1	Introduce el nombre de usuario y contraseña, que dio de alta el administrador del sistema.		
2		Valida los datos del usuario, si son correctos el sistema permite el ingreso.	En caso de que alguno de los datos no sea correcto se emite el mensaje.

Caso de uso: Registrar Producto IDCU: 2			
Paso	Usuario	Sistema	Alterno
1	Llena los campos correspondientes a la descripción del producto y da clic en el botón de guardar.		
2		Valida los datos y guarda en la base de datos, si ya existe un registro igual mandara el mensaje indicando ésta situación.	Si los datos son correctos se guardan en la base de datos y el sistema confirma que se guardo el registro.

Caso de uso: Registrar Troquel IDCU: 3			
Paso	Usuario	Sistema	Alterno
1	Llena los campos del formulario Y al terminar da clic en el botón guardar.		
2		Verifica que estén llenos los campos que son necesarios para crear el código. Verifica también que el código del troquel no se esté repitiendo.	Si faltan campos para crear el código emite un mensaje señalando ésta situación. Si ya existe el código se da el aviso de modificar el número de herramienta.
3	Si el sistema no le devuelve ningún mensaje, elige las partes del troquel y las guarda. De no ser así completa los campos faltantes o incrementa el número de herramienta hasta completarse el código.		
4		Guarda los datos en la tabla Troquel y la tabla Partes confirmando que guardó el registro.	

Caso de uso: Modificar y/o Eliminar registros IDCU: 4			
Paso	Usuario	Sistema	Alterno
1	Selecciona el registro que desea modificar, escribe el nuevo dato en el campo correspondiente y da clic en el botón modificar registro.		
2		Si el campo no está restringido permite la modificación y la guarda.	
3	Si lo que desea es eliminar un registro deberá seleccionarlo y dar clic en el botón eliminar registro.		
4		Actualiza la base de datos después de haber eliminado un registro.	No permite crear nuevos registros de troqueles solo agregar partes del troquel.

Caso de Uso: Búsqueda de información IDCU: 5			
Paso	Usuario	Sistema	Alterno
1	Elige la tabla, el campo y teclea el dato que quiere buscar, da clic en el botón buscar.		
2		Muestra la información correspondiente a los datos proporcionados.	Si no existe información o el campo de búsqueda no es correcto manda un mensaje de aviso.

Caso de Uso: Búsqueda por similitud IDCU: 6			
Paso	Usuario	Sistema	Alterno
1	Introduce las características del nuevo producto y elige asignar índices.	Realiza la asignación de los índices que serán comparados.	
2	Inicia la comparación de atributos dando clic en el botón buscar.		
3		Aplica el algoritmo de RBC y muestra los resultados por porcentajes de similitud con productos guardados en la base de casos.	Si la base de casos no tiene similitudes lo indica por medio de un mensaje.
4	Elige los resultados que tengan mayor similitud.		
5		Despliega la información correspondiente a la solución del producto en donde se proporciona la lista de troqueles que se utilizaron.	Imprime en papel los datos de la solución.

Caso de Uso:Recupera y adapta la solución IDCU: 7			
Paso	Usuario	Sistema	Alterno
1	Visualiza los resultados y elige los troqueles que le sean utiles. Las características que le falten del producto las busca nuevamente para tener una solución completa.		
2		Muestra la información de la base de datos relacionada con la característica que el usuario requiere.	
3	Adapta los resultados a la solución, si es completa se guarda como un nuevo caso.		
4		Guarda el registro	

La figura 23 muestra el diagrama de casos de uso para las actividades del administrador del sistema.

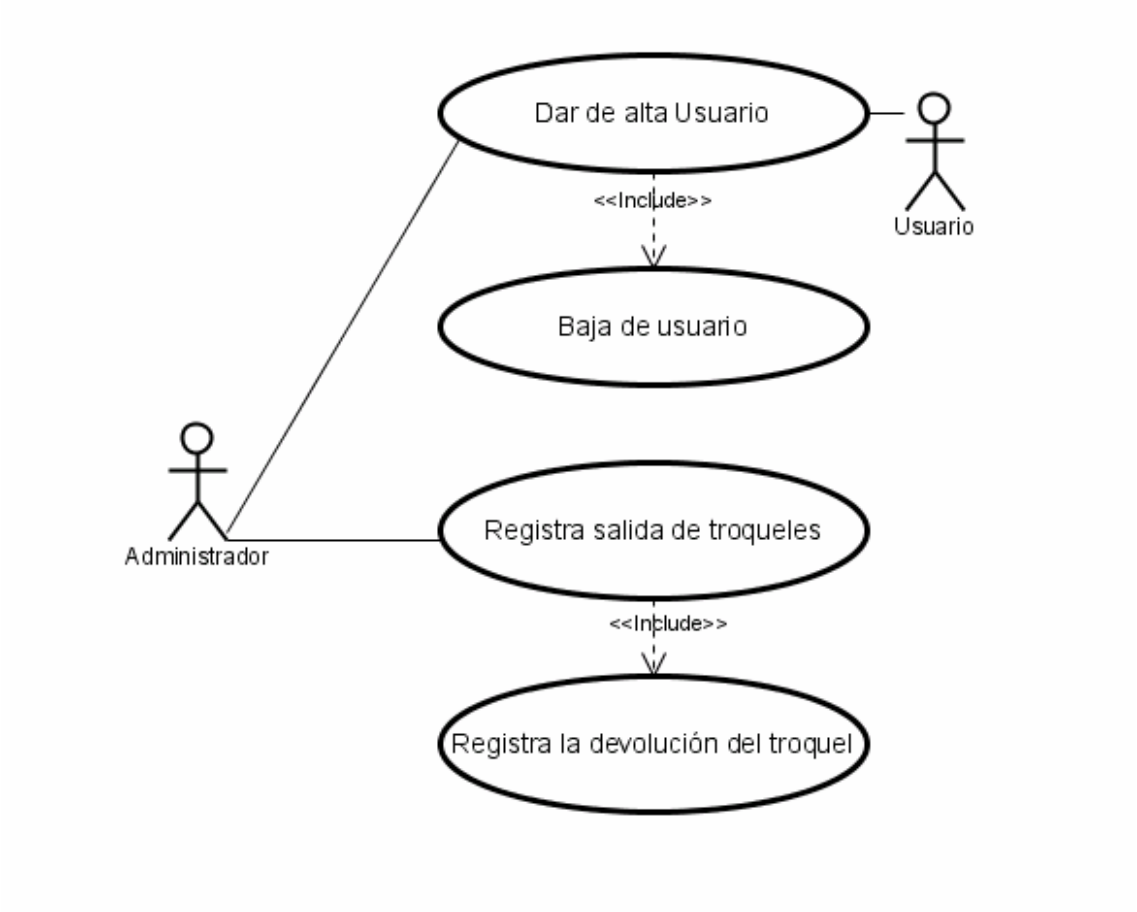


Figura 23. Casos de uso para el administrador del sistema

Caso de uso: Dar de alta Usuario			
Paso	Administrador	Sistema	Alterno
1	Introduce los datos del usuario y da clic en guardar.		
2		Almacena la información en la base de datos y envía un mensaje confirmando.	Si existe un usuario ó contraseña igual no permite guardar los datos y manda un mensaje señalando este problema.
3	Realiza los cambios y guarda los datos.	Almacena la información en la base de datos y envía un mensaje confirmando.	

Caso de uso: Baja de Usuario			
Paso	Administrador	Sistema	Alterno
1	Introduce el nombre de usuario y da clic en baja de usuario.		
2		Elimina el registro en la base de datos correspondiente al usuario.	

Caso de uso: Registra salida de troqueles			
Paso	Administrador	Sistema	Alterno
1	Llena el registro de los troqueles que salen del almacén.		
2		Guarda el registro y modifica la ubicación del troquel en la tabla de troqueles.	

Caso de uso: Registra devolución de troqueles			
Paso	Administrador	Sistema	Alterno
1	Registra la fecha en la que se devuelve el troquel.		
2		Guarda el registro y modifica la ubicación del troquel en la tabla de troqueles.	

Capítulo 6. Desarrollo.

Nuestra base de datos es de tipo relacional, se manejan varias tablas bajo una misma identificación que trabajan en base a relaciones entre las mismas, las relaciones pueden ser entre dos o más tablas y puede generarse una nueva a partir de los registros que cumplen con el criterio de correspondencia, las relaciones se llevan a cabo a través de campos llaves.

Una relación es la correspondencia existente entre los registros de las diferentes tablas que componen la base de datos relacional.

6.1 Desarrollo de la base de datos

Se establecen las relaciones de las tablas:

El tipo de entidad **Familia** participa en la siguiente interrelación:
Familia/BaseDeCasos: tipo de interrelación 1: N

El tipo de entidad **Geometría** participa en la siguiente interrelación:
Geometría/BaseDeCasos: tipo de interrelación 1: N

El tipo de entidad **BaseDeCasos** participa en la siguiente interrelación:
BaseDeCasos/Troquel: tipo de interrelación 1: N
BaseDeCasos/RelacionComponente: tipo de interrelación 1: N

El tipo de entidad **Troquel** participa en la siguiente interrelación:
Troquel/Partes: tipo de interrelación 1: N

El tipo de entidad **Operaciones** participa en la siguiente interrelación:
Operaciones/PartesxOp: tipo de interrelación 1: N

En la figura 24 se muestra el diagrama de relaciones de la base de datos:

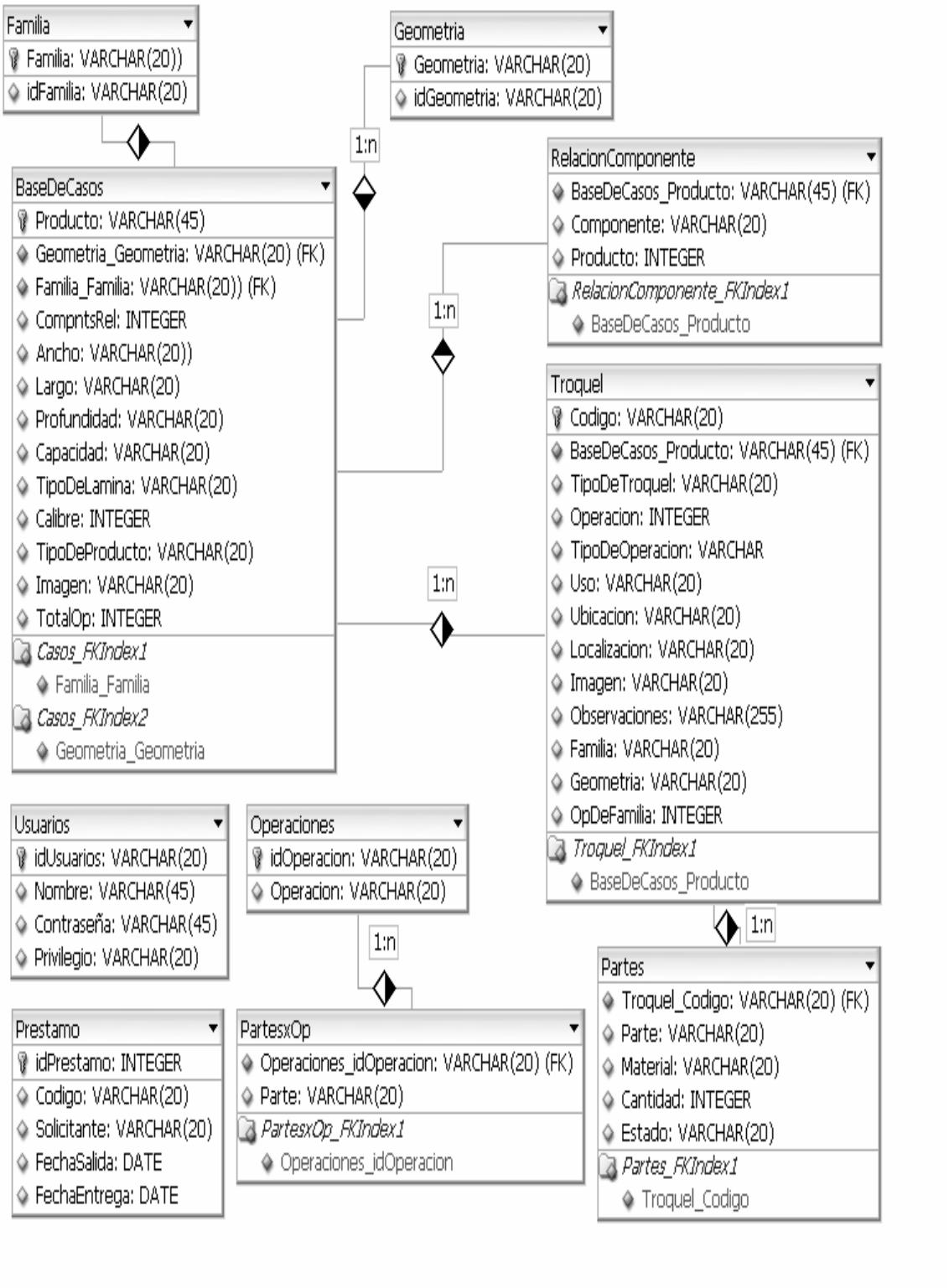


Figura 24. Diagrama relacional de la base de datos.

6.2 Desarrollo de la interfaz del sistema

Mediante el diagrama de actividades se representa el camino que seguirá el flujo de información.

Las herramientas de programación que se emplean en el desarrollo de las aplicaciones son:

Visual Basic.net con la cual se creara la interfaz de usuario y se llevara a cabo la conexión con la base de datos mediante el modelo de acceso a datos ADO.net

El manejador de bases de datos es Access en el que se administrara la base de datos BDGZ.

Otra herramienta que se emplea es el lenguaje de consulta de base de datos SQL, que permite manipular los datos de la base mediante instrucciones que clasifican, ordenan y muestran la información en diferentes formatos.

En cada actividad se determinan diferentes instrucciones, manejo de clases, objetos, construcciones para el control de flujo de datos, funciones y procedimientos.

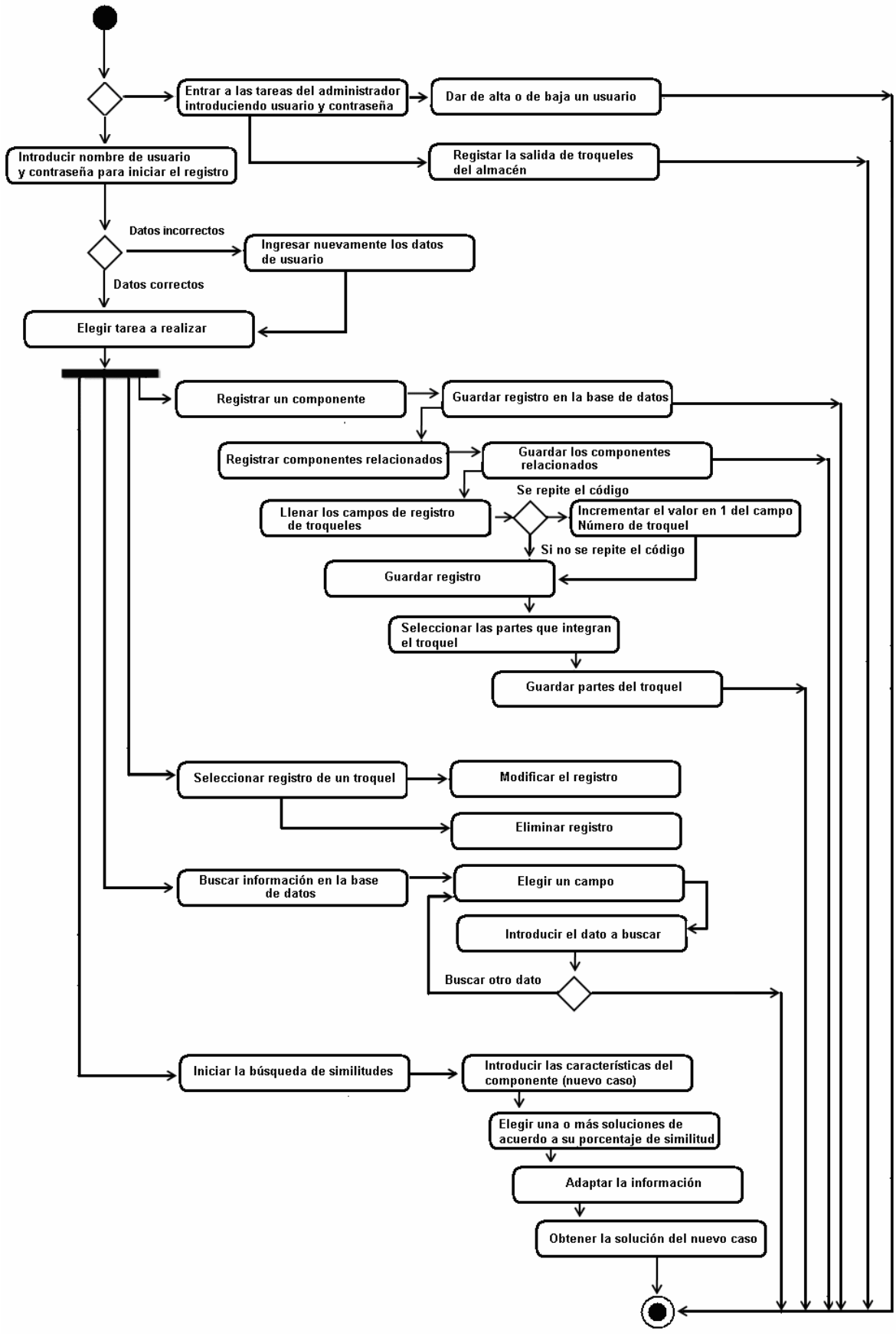


Figura 25. Diagrama de actividades del sistema

Actividad: Introducir nombre de usuario y contraseña para iniciar la aplicación.

Datos de entrada: Nombre de Usuario y contraseña

1.- Comparación de la cadena de caracteres con los campos de la tabla “Usuarios”, de acuerdo al resultado de la comparación se permite el acceso al “Menú” donde aparecerá en primera instancia la pantalla de “Registro de productos”. Figura 26

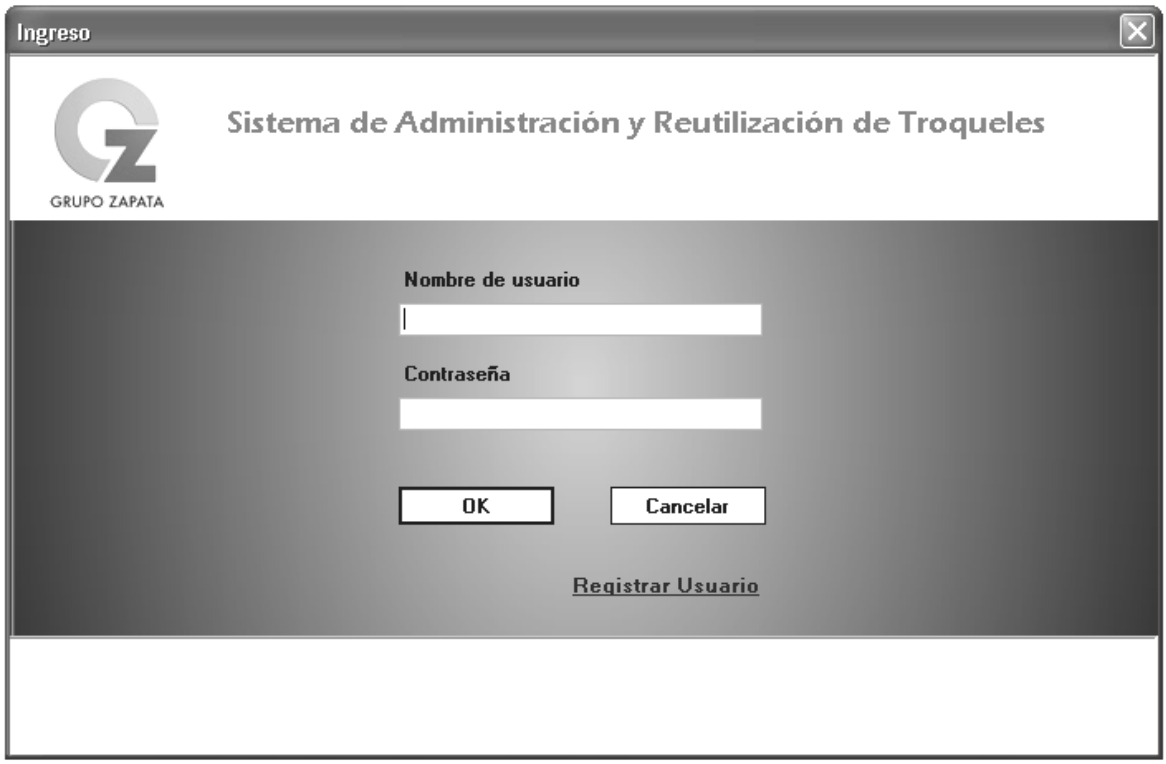


Figura 26. Pantalla de acceso a las aplicaciones del sistema

Si la comparación de los datos devuelve que los datos no son correctos, se ingresa nuevamente los datos, y si nuevamente no son correctos el usuario debe verificar sus datos con la persona encargada del sistema.

Actividad: Elegir una tarea a realizar
Datos de entrada: clic sobre el botón que describa la opción.

1.- La pantalla presenta botones con las siguientes opciones:
Registrar medida nominal
Registrar troquel
Modificar o eliminar troquel
Búsqueda de información
Búsqueda por casos

2. Al elegir una opción se lleva a cabo un evento el cual provoca que se muestre la Pantalla elegida.

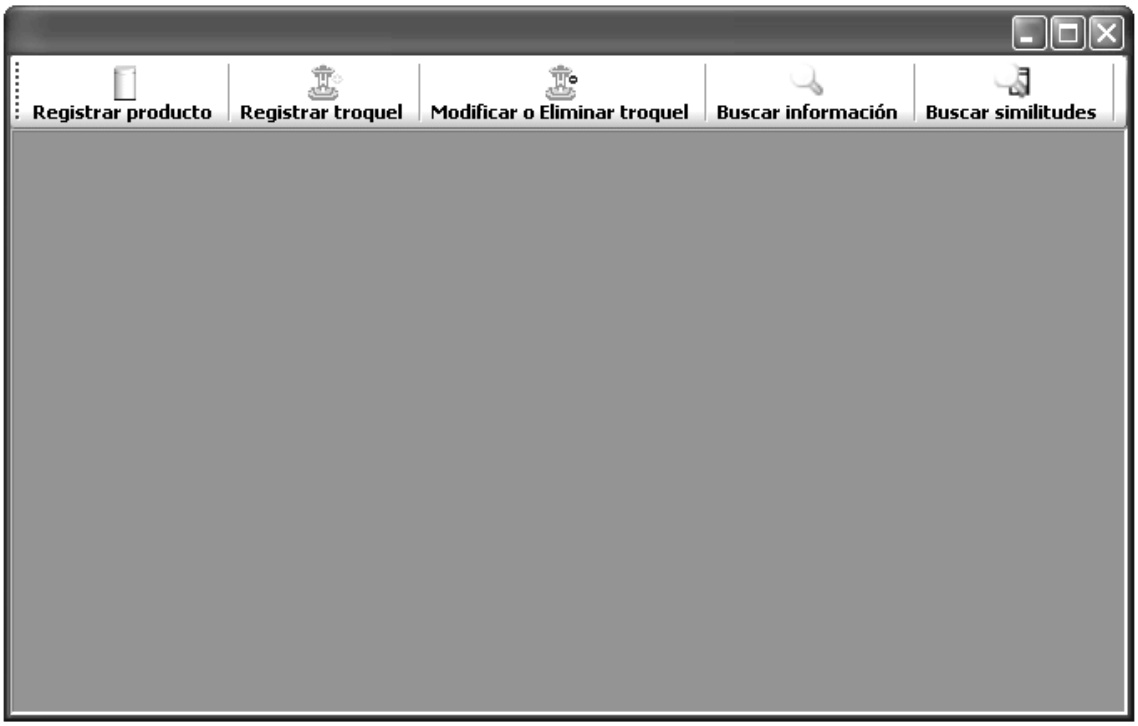


Figura 27. Pantalla de menú de opciones

Actividad: Registrar un producto
Datos de entrada: El usuario llena los formularios eligiendo los datos de una lista desplegada, en algunos campos, y tecleando el dato en las cajas de texto en otros campos.

1. La pantalla se encuentra conectada a la base de datos con las tablas “BaseDeCasos” “Familia”, “Geometria” y “RelacionDeComponentes” de acuerdo a lo establecido en el diagrama de relaciones de la base de datos las tablas se interrelacionan.

2. De la tabla “Familia” y “Geometria” leemos los datos para elegirlos en el campo correspondiente.

3. Los datos introducidos son guardados en la tabla “BaseDeCasos”, si existen componentes relacionados con el producto se abre un campo en el cual se escribe el nombre del componente y éste se guarda en la tabla “RelacionDeComponentes”

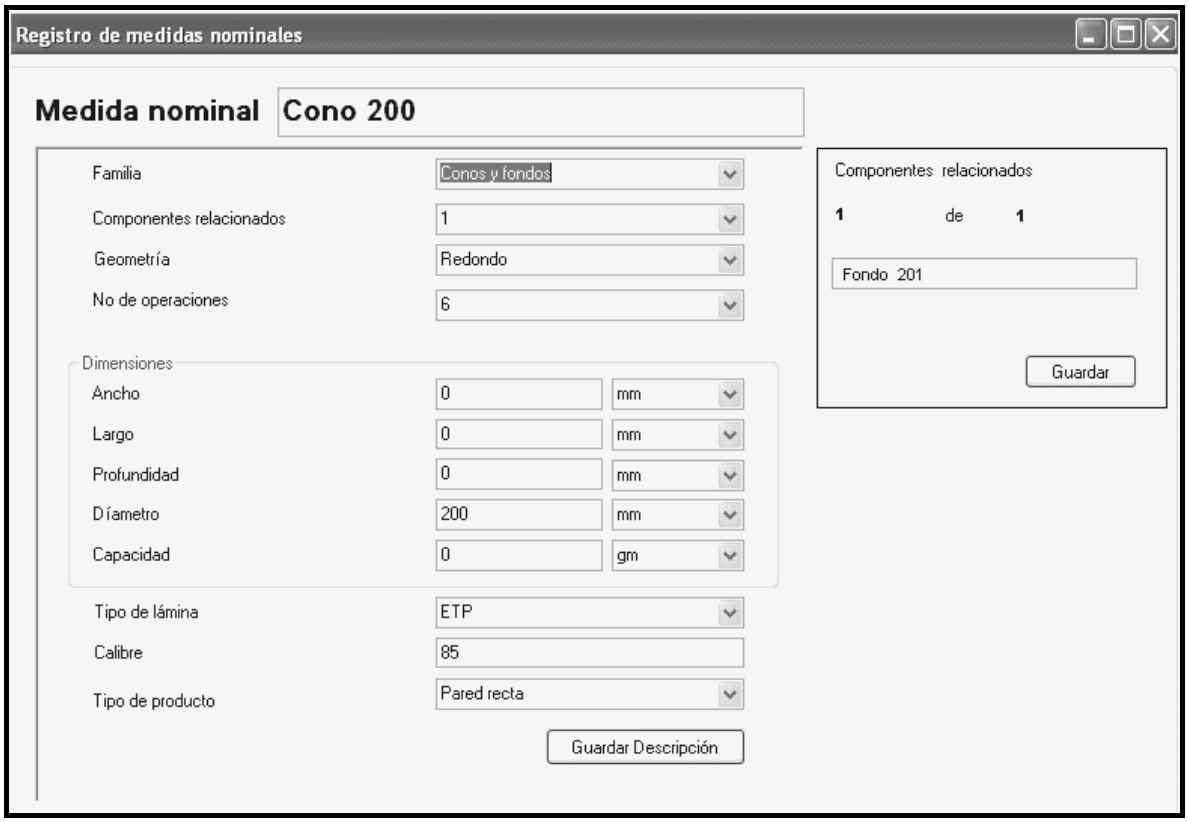


Figura 28. Pantalla de registro de productos

Actividad: Llenar los campos de registros de troqueles
Datos de entrada: el usuario llena solamente los campos habilitados para escritura.

1. Esta pantalla esta conectada a la base de datos con las tablas “Troquel”, “Partes”, “BasedeCasos”.

2. Al elegir el producto que realiza el troquel aparecen los datos de la familia, las operaciones totales para realizar el producto y su geometría, esto define parte del código de identificación del troquel que se irá creando conforme el usuario elija las opciones que describan de mejor manera el troquel.

3. Al completarse el registro se guarda dando clic en el botón guardar, el evento de este botón manda los datos a la tabla “Troquel”, pero antes de eso en el campo “Número de troquel” hay un evento que manda a comparar el código que se ha creado con los códigos ya registrados anteriormente, de existir uno igual se emite un mensaje informando la situación y se procede por parte del usuario a incrementar el campo “Número de troquel” en 1 dígito más.

4. El registro de partes depende de dos cosas:
a) que al elegir en el campo “Tipo de operación” una opción, se llene la lista de partes especificas y
b) que se haya guardado el registro del troquel, de otra forma no se habilitara el listado y no se podrá elegir una parte.

En caso de que al elegir un tipo de operación no se despliegue la lista de partes se puede seleccionar “Partes generales” con este evento se desplegara una lista de partes posibles en un troquel.

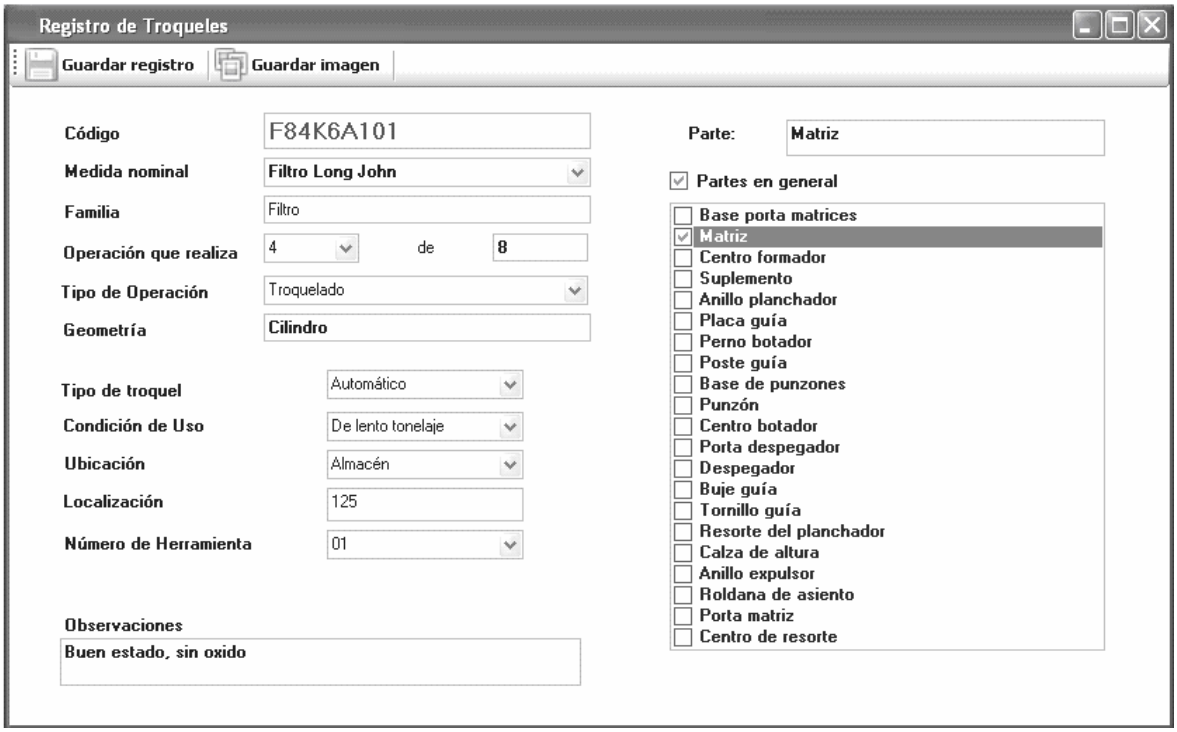


Figura 29. Registro de troqueles

Actividad: Modificar o eliminar registros de troquel

1 Al elegir un código de la lista se despliega su información, es posible editar su contenido si el campo no está restringido, el usuario teclea el nuevo dato y se guarda la modificación al dar clic en el botón correspondiente cuyo evento realiza el cambio en la base de datos. Se pueden agregar o quitar partes del troquel en el caso de ser necesario.

2. Para eliminar un registro el usuario debe seleccionar el código de la lista y desplegar su información, al oprimir el botón “Eliminar registro” el evento realiza la instrucción Delate para borrar los registros de la tabla “Troquel” y sus correspondientes partes en la tabla “Partes”. Se actualiza la tabla.

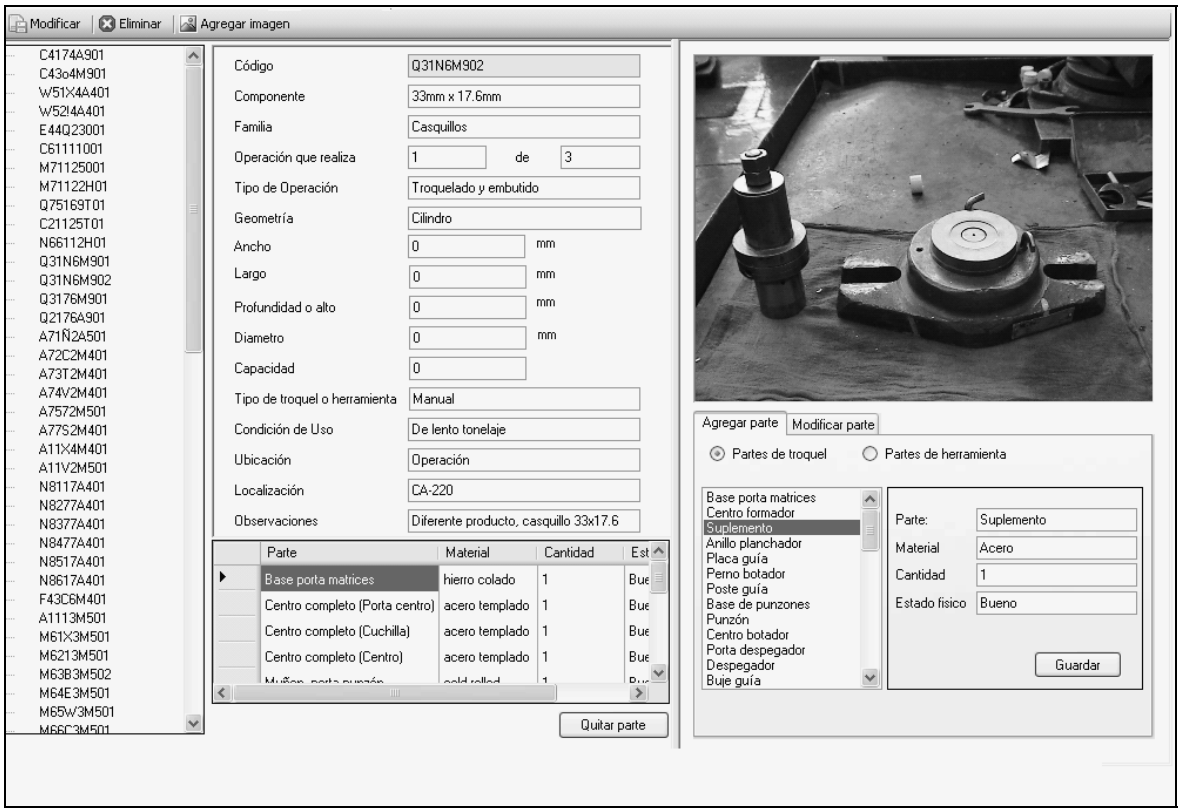


Figura 30. Pantalla para modificar o eliminar registros de troqueles

Actividad: Buscar información

1. Las opciones disponibles en la pantalla manejan datos pertenecientes a la base de datos del sistema que mediante instrucciones SQL realizan un filtro de la información correspondiente a lo que indica el usuario al elegir un miembro de cada opción.

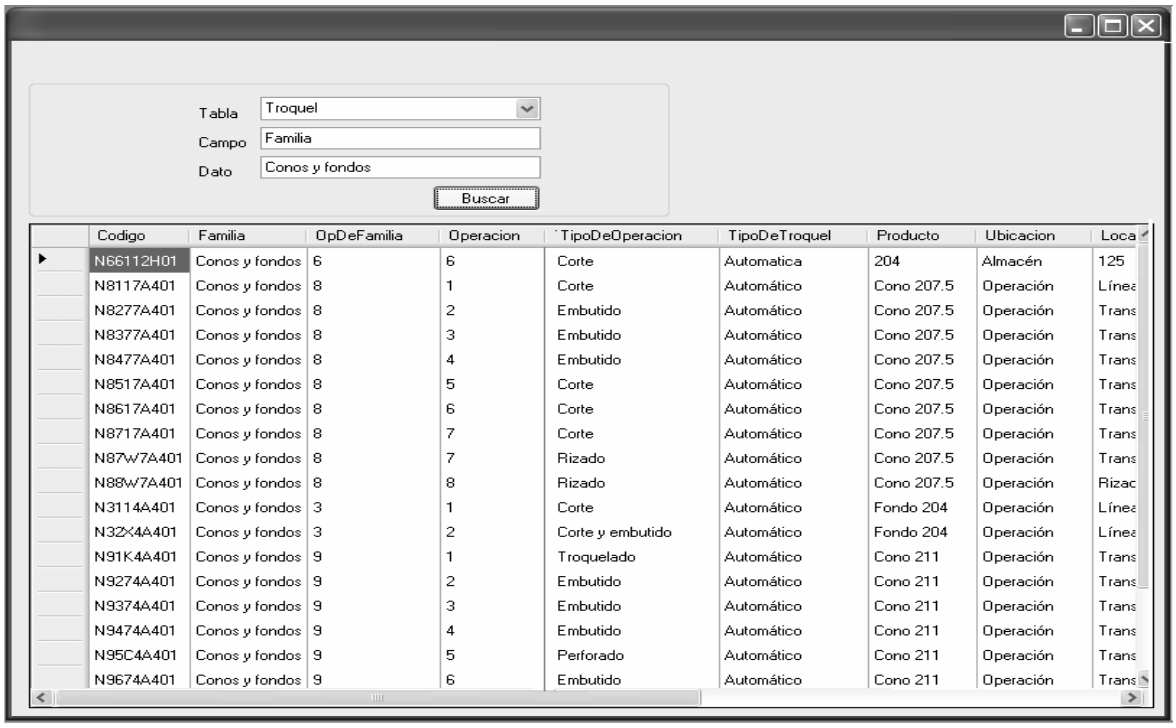


Figura 31. Búsqueda de información

Actividades para el administrador: Entrar a las tareas de administrador

Datos de entrada: Nombre de Usuario con privilegios de administrador y contraseña

Al entrar los datos se hará la validación del nombre de Usuario su contraseña y el privilegio de administrador que debe tener

Figura 32. Acceso a las tareas de administrador

Actividad: Registrar usuario
Datos de entrada: Nombre completo, nombre de usuario, contraseña y privilegio

1. Al llenar los campos con los datos de entrada el sistema guarda la información en la tabla “Usuarios” de la base de datos

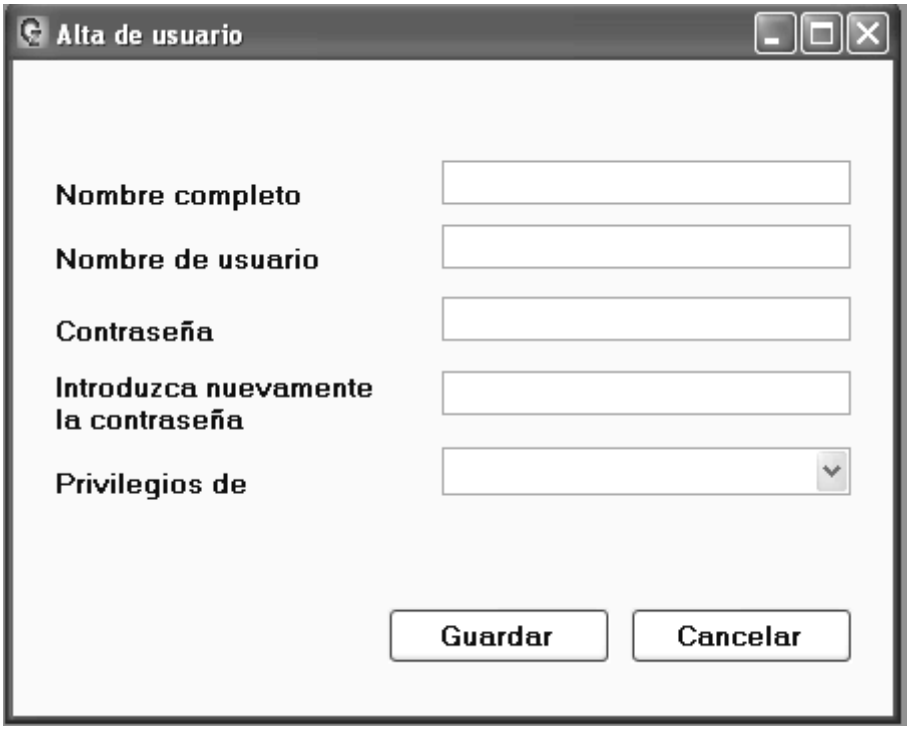


Figura 33. Pantalla de alta de usuario del sistema

Actividad: Baja de usuario
Dato de entrada: Nombre de usuario
1. Da de baja el registro al dar clic en el botón OK y actualiza la tabla de usuarios.

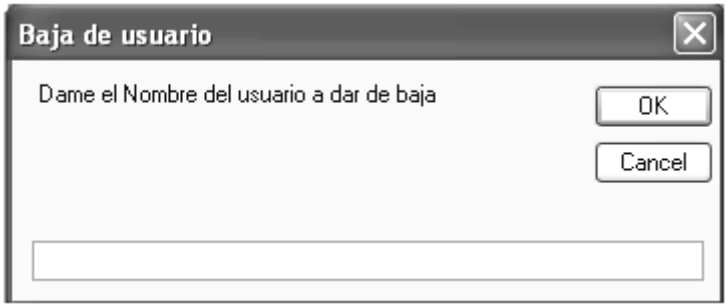


Figura 34. Baja de usuario

Actividad: Préstamo de troqueles
Datos de entrada: Nombre de quien solicita el troquel, código del troquel, fecha de salida y ubicación.

1. Se guardan los datos en la tabla “Préstamo” mediante el botón guardar

Figura 35. Control de la salida de troqueles

6.3 Desarrollo del algoritmo de razonamiento basado en casos.

Datos de entrada:

Atributo	Tipo de dato
Familia	Cadena de caracteres
Elementos relacionados	Entero
Geometría	Cadena de caracteres
No de Operaciones	Entero
Ancho	Entero
Largo	Entero
Profundidad	Entero
Diámetro	Entero
Capacidad	Entero
Tipo de lámina	Cadena de caracteres
Calibre	Entero
Tipo de producto	Cadena de caracteres

Tabla 7. Argumentos para la búsqueda

Asignación de índices:

Estos son los atributos a los cuales serán asignados valores que dentro de los términos del algoritmo de razonamiento basado en casos son los índices que nos permiten recuperar los casos previos.

Los datos de entrada son guardados en un arreglo, para posteriormente compararlo con un arreglo que contiene los atributos de los casos previos.

Recuperación

Los datos de la tabla “BaseDeCasos” son guardados en un arreglo que se compara con el arreglo de los atributos del nuevo caso, de esa comparación se obtienen los valores de distancia (0,1 ó 0.5) estos son multiplicados por los elementos de pesos que le corresponden a cada atributo, se hace una suma de los resultados que se divide entre la suma de todos los pesos con lo que se obtiene un valor que será el porcentaje de similitud entre el caso nuevo y el caso previo. Figura 36

Se comparan todos los atributos de cada registro con los atributos del caso nuevo por medio de una instrucción de control de flujo (for - next) desde el primer registro hasta el último.

Los resultados deben aparecer de manera ordenada del porcentaje más alto hasta el más bajo

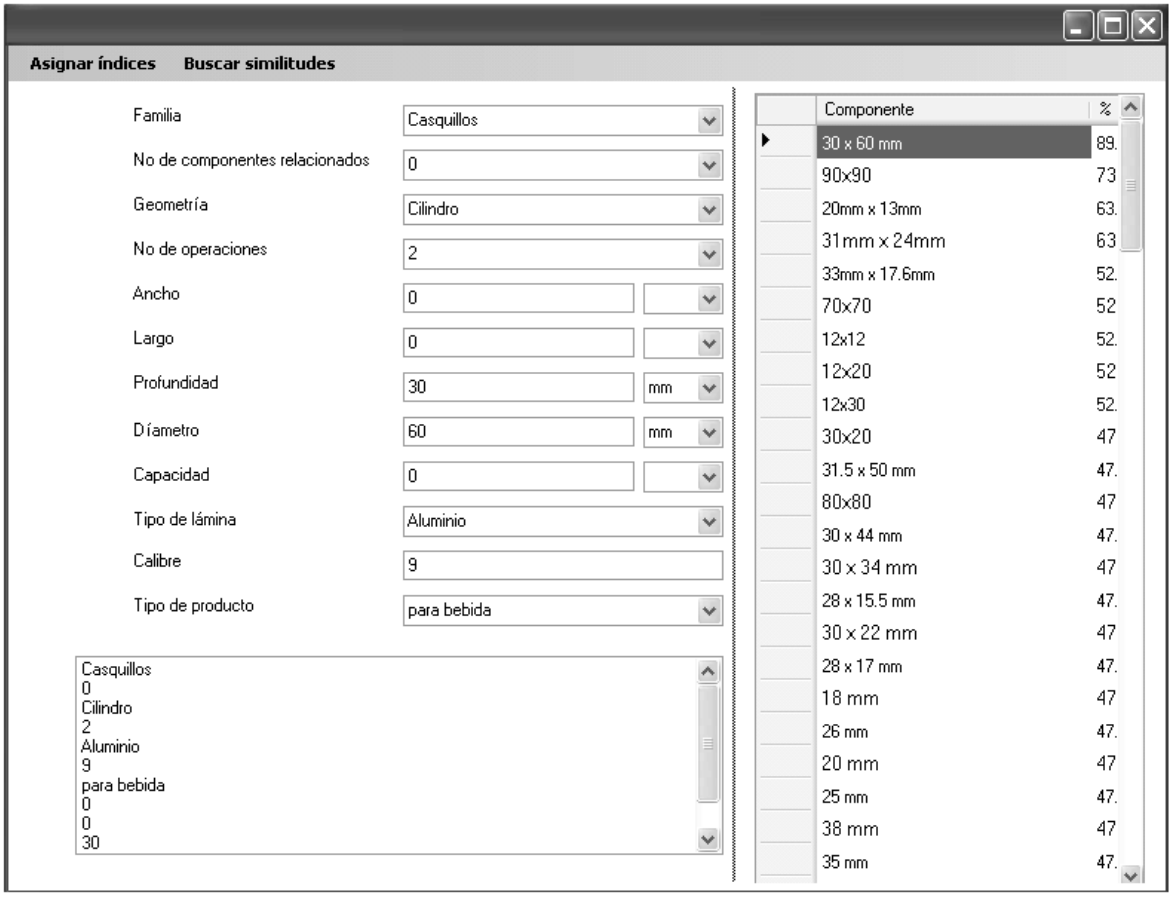


Figura 36. Pantalla donde se realiza la asignación de índices y la búsqueda de similitudes

El usuario elige uno de los resultados para revisar la solución y se desplegara una pantalla que contiene la descripción del caso, los troqueles que son usados, se describen sus características y las partes que lo componen. Figura 37.

Adaptación.

La solución presenta también las diferencias que hay entre los casos lo cual permite hacer las adaptaciones necesarias para resolver el problema.

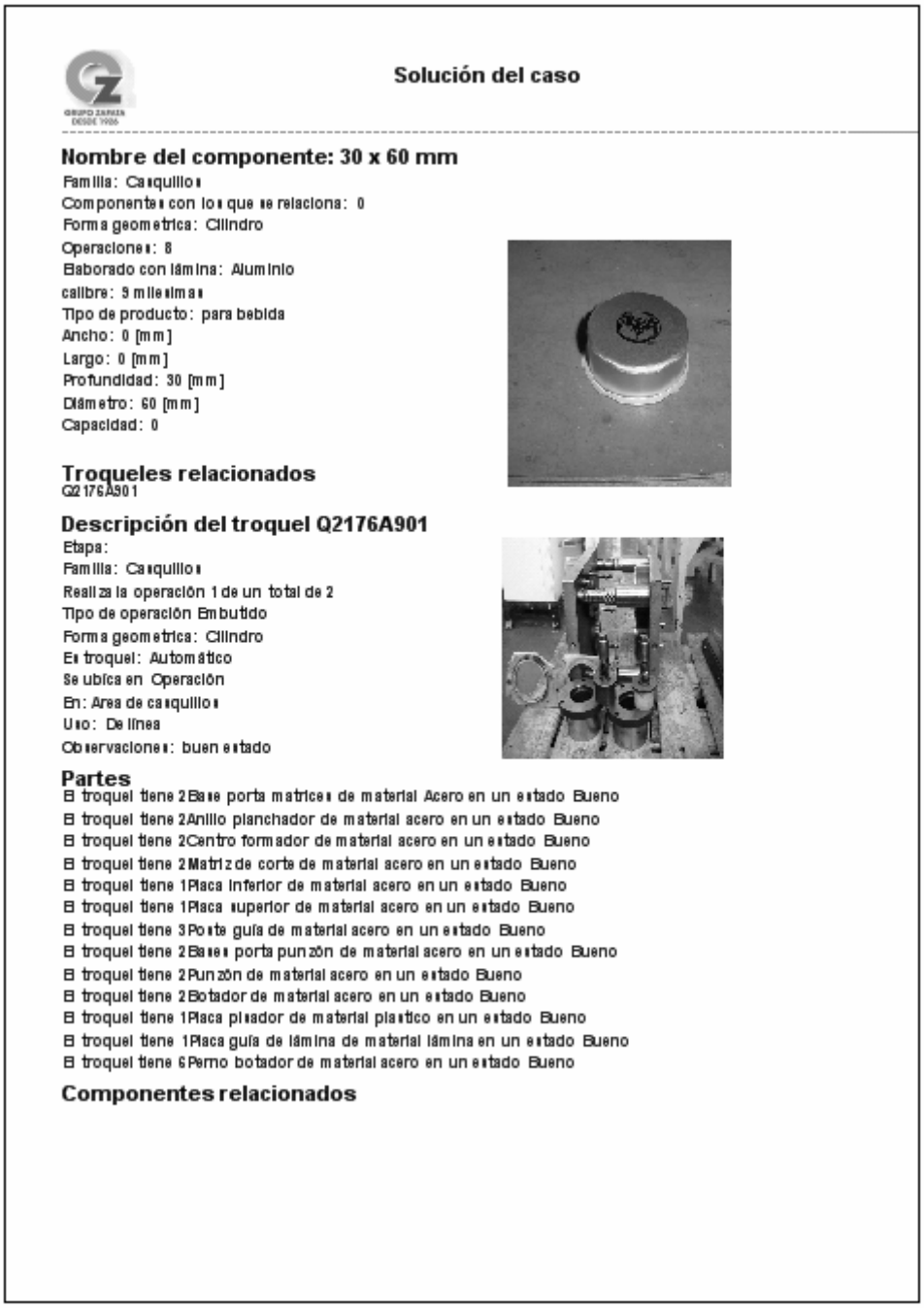


Figura 37. Pantalla de solución después de elegir un caso resultante

Capítulo 7. Validación y verificación del sistema.

La validación del software se consigue mediante una serie de pruebas que demuestran la conformidad con los requisitos. Se determina un plan de prueba, el cual consiste en trazar las pruebas que se han de llevar a cabo y un procedimiento de pruebas define los casos de prueba que se usarán para demostrar la conformidad con los requisitos. Tanto el plan como el procedimiento estarán diseñados para asegurar que se satisfacen los requisitos funcionales, que se alcanzan todos los requisitos de rendimiento, que la documentación es correcta.

7.1 Casos de prueba

Caso de prueba:	Ingreso al sistema
Propósito:	Tener control del ingreso de los usuarios con una contraseña
Prerrequisitos:	Contar con un registro de usuario asignado previamente por el administrador del sistema
Datos de prueba:	Nombre de usuario={usuario válido, usuario inválido, vacío} Contraseña={válido, inválido, vacío}
Pasos:	1. Entrar a la página de acceso de usuarios. 2. Teclear Nombre de Usuario 3. Teclear Contraseña 4. Hacer clic en entrar
Finalización:	Cancelar
Resultados esperados:	a) Usuario y Contraseña válidos permite el ingreso a la pantalla Registro de medidas nominales b) Usuario válido y Contraseña inválida o vacía, ingreso no permitido y el sistema emite un mensaje c) Usuario inválido o vacío y Contraseña válida, ingreso no permitido y el sistema emite un mensaje d) Usuario no válido o vacío y Contraseña no válida o vacío, ingreso no permitido y el sistema emite un mensaje

Caso de prueba:	Registrar y guardar medida nominal
Propósito:	Tener registros completos de los productos
Prerrequisitos:	Ninguno
Datos de prueba:	Cadenas (alfanuméricas, con caracteres especiales y vacía), Números (entero en rango positivo, decimales)
	Lista de opciones (una opción o ninguna)
Pasos:	1. Teclear en el campo medida nominal 2. Llenar todos los campos 3. Hacer clic en Guardar
Finalización:	Guardar elementos relacionados con el producto
Resultados esperados: a) Todos los campos llenos se guarda el registro y se envía un mensaje señalando la acción b) Campo "Medida nominal" vacío, el registro no se guarda y se emite mensaje para completar los campos c) Alguno de los campos esta vacío, el registro no se guarda y se emite mensaje para completar los campos	

Caso de prueba:	Registrar y guardar información de troqueles.
Propósito:	Verificar la creación del código de identificación y llenar los campos necesarios para esto, guardar los registros en la base de datos.
Datos de prueba:	Cadenas (alfanuméricas, vacía) Números(Enteros y decimales positivos) Listas de opciones
Prerrequisitos:	Tener guardados registros de productos
Pasos:	1. Elegir producto 2. Elegir componente 3. Elegir operación que realiza el troquel 4. Elegir tipo de operación 5. Elegir el número de herramienta 6. guardar
Resultados esperados:	a) Se genera el código si están llenos los campos que lo crean, si falta algún campo no se puede completar el código y no se podrá guardar el registro. b) Si el código esta completo y el sistema no emite ningún mensaje el registro esta listo para guardarlo c) Si el código ya existe el sistema emite un mensaje para modificar el campo "número de troquel" y el registro no se puede guardar hasta realizar este cambio. d) Al guardar el registro se habilita el área de partes

Caso de prueba:	Eliminar o modificar registros de los troqueles
Propósito:	Comprobar que los registros que ya no sean necesarios desaparecen de la base de datos así como actualizar los registros modificados.
Datos de prueba:	Cadenas(vacía, alfanumérica, caracteres especiales), Números(enteros, decimales, 0)
Prerrequisitos:	Tener registros de troqueles.
Pasos:	1. Seleccionar el registro que se quiere modificar o eliminar 2. Si es una eliminación hacer clic en "Eliminar" 3. Si lo que se quiere es modificar se elige el código del troquel correspondiente. 4. Teclear nuevo dato 5. Hacer clic en "Guardar modificación"
Resultados Esperados:	a) Al seleccionar el código en la lista, se debe desplegar la información en cada parte del formulario. b) Al eliminarse un registro debe desaparecer de la lista, la cual muestra lo que hay en la base de datos por lo tanto el hecho de no aparecer quiere decir que se actualizó la base de datos. c) Al modificar los campos permitidos del registro seleccionado, se emitirá un mensaje de aviso.

Caso de prueba:	Buscar información en las tablas de la base de datos.
Propósito:	Ver los resultados correctos de acuerdo al dato dado por el usuario
Datos de prueba:	Cadenas(vacía, alfanumérica, caracteres especiales) Números(enteros, decimales, 0)
Prerrequisitos:	Contener información en la base de datos.
Pasos:	1. Elegir una opción del grupo tablas 2. Si la elección es buscar por campo se elige uno 3. Teclear un dato 4. Hacer clic en el botón "Buscar" 5. Ver resultados
Resultados esperados:	a) Al elegir una opción del grupo de tablas, se desplegará la información contenida en ella. b) Si los datos que se introducen son vacíos no hay búsqueda y el sistema pedirá que teclee un dato. c) Se desplegarán los resultados en la pantalla si existen registros con el dato buscado, si no existe alguno el sistema mandará un mensaje señalando este hecho.

Caso de prueba:	Buscar casos de similitud
Propósito:	Comprobar el funcionamiento del algoritmo de razonamiento basado en casos
Datos de prueba:	Cadenas(vacía, alfanumérica, caracteres especiales), números (enteros, decimales, 0)
Prerrequisitos:	Contener registros de los productos.
Pasos:	1. Llenar todos los campos 2. Presentar el nuevo caso 3. Hacer clic en "buscar similitudes" 4. Elegir una solución para ver su contenido
Resultados esperados:	a) Si todos los campos están llenos se puede realizar la búsqueda. b) Si algún campo recibe un dato vacío se emitirá un mensaje al presentar el nuevo caso señalando la acción. c) Los resultados deberán presentarse en orden descendente de acuerdo al porcentaje de similitud del caso nuevo con los casos en la base de conocimiento. d) Al seleccionar un resultado se abrirá una página en la cual se encontrará toda la información de los troqueles con los que se fabrica el producto.

7.2 Pruebas de usuario

Prueba de integración:

La prueba de integración tiene como objetivo verificar que todos los componentes del sistema interactúen de acuerdo al esquema de actividades y conexiones con la base de datos.

Acciones:

1. Verificar las conexiones de cada formulario con la base de datos a la correspondiente tabla donde guardarán datos o modificaran registros.
2. Verificar las conexiones con la base de datos para las tablas de datos donde se deben borrar registros
3. Verificar los eventos donde se pretende vincular las pantallas.

Prueba de volumen:

La prueba de volumen pretende observar el comportamiento del sistema frente a la captura masiva de los registros, así como de las tareas de modificación y eliminación.

Capítulo 8. Análisis de resultados.

8.1. Conclusiones.

Los resultados esperados en los casos de pruebas fueron en su mayoría satisfactorios, corrigiéndose algunos detalles que no permitían la integración total de los demás bloques y la interacción con el usuario.

Al terminar las correcciones los resultados fueron completamente satisfactorios lo que implica que se cumplió con los requerimientos planteados por el cliente.

La técnica de razonamiento basado en casos, proporciono elementos suficientes para poder encontrar soluciones que ayuden con la tarea principal que es ahorrar tiempo y dinero a la empresa.

Ya que el cliente durante el desarrollo del sistema solicitó varios cambios, la técnica de RBC en un principio contemplaba trabajar con una cantidad considerable de clases y atributos, los cambios propiciaron la reducción de elementos, pese ha esto siguió siendo útil en las modificaciones.

Se destaca la importancia que tiene la recopilación de información y la cooperación que el cliente tenga al hacer el levantamiento de requerimientos ya que hubo en ocasiones poca disponibilidad, esto implicó retrasos de acuerdo a los tiempos que se establecieron en la propuesta inicial.

Durante las reuniones en donde se veían los avances del sistema se pedían cambios que en un momento dado implicó replantear los requerimientos y modificar el diseño, que aunque la metodología en espiral permite regresar a pasos anteriores, esto afectaba a otras actividades relacionadas con la terminación del sistema.

El sistema tiene la capacidad de crecer por lo que es importante mejorar en los aspectos antes mencionados.

Anexo

Sistema de Administración y Reutilización de Troqueles
Código fuente de las aplicaciones

Pantalla: Ingreso a las aplicaciones

```
             'Comparación de la cadena de caracteres  con los campos de la
             'tabla “Usuarios”,  de acuerdo al resultado de la comparación se
             'permite  el acceso al “Menú” donde aparecerá en primera
             'instancia la pantalla de “Registro de productos”
Imports System.Data.OleDb
Public Class Ingreso
    Private oDataAdapter As OleDbDataAdapter
    Private oDataSet As DataSet
    Private oDataRow As DataRow
    Private ipos As Integer

    Private Sub OK_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles OK.Click
        Try
            'conexion a la base de datos
            Dim oConexion As OleDbConnection
            oConexion = New
            'compara los datos de entrada del usuario
OleDbConnection("provider=Microsoft.jet.oledb.4.0;Data source=BDGZ.mdb;")
            Me.oDataAdapter = New OleDbDataAdapter("select * from Usuarios
where idUsuario like '" & Me.TxtUsuario.Text & "' and  Contraseña like '" &
Me.TxtContraseña.Text & "'", oConexion)
            Me.oDataSet = New DataSet

            'se abre conexion con la base usuario
            oConexion.Open()

            'se llena el DataAdapter con los datos
            Me.oDataAdapter.Fill(oDataSet, "Usuarios")
            ' se cierra la conexion
            oConexion.Close()
            Me.ipos = 0
            Me.Carga()
            'Me.Close()

            Me.txtContraseña.Clear()
            Me.TxtUsuario.Clear()
        Catch ex As Exception
            'si los datos no son válidos genera éste mensaje
            MessageBox.Show("Usuario o Contraseña no validos", "",
MessageBoxButtons.OK)
            Me.TxtUsuario.Clear()
            Me.TxtContraseña.Clear()
        End Try
    End Sub
    Private Sub Carga()
        oDataRow = Me.oDataSet.Tables("Usuarios").Rows(Me.ipos)
        Dim vmenu As New Menu
        vmenu.Show()
    End Sub

    Private Sub Cancel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Cancel.Click
        Me.Close()
    End Sub
       'vínculo a las tareas de administrador
    Private Sub LinkRegistrarU_LinkClicked(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles
LinkRegistrarU.LinkClicked
```

```
        Dim vloginadm As New LoginAdm
        vloginadm.Show()
    End Sub

    Private Sub Ingreso_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub
End Class
```

Pantalla: Menú de aplicaciones

```
Public Class Menu
‘ En cada opción se crea el vínculo para abrir las diferentes aplicaciones
    Private Sub TbtnGC_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TbtnGC.Click
        Dim frmVentanaDentro As New GuardarCaso
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Registro productos"
        frmVentanaDentro.Show()

    End Sub

    Private Sub ToolStripButton2_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ToolStripButton2.Click
        Dim frmVentanaDentro As New RegistarTroqueles
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Registro de Troqueles"
        frmVentanaDentro.Show()
    End Sub

    Private Sub ToolStripButton3_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ToolStripButton3.Click
        Dim frmVentanaDentro As New Buscar
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Búsqueda"
        frmVentanaDentro.Show()
    End Sub

    Private Sub tsbtnModificar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles tsbtnModificar.Click
        Dim frmVentanaDentro As New Modificar
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Modificar"
        frmVentanaDentro.Show()
    End Sub


    Private Sub ToolStripButton4_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ToolStripButton4.Click
        Dim frmVentanaDentro As New rbcN
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Buscar similitudes"
        frmVentanaDentro.Show()
    End Sub

End Class
```

Pantalla: GuardarCaso

```
            'Se llenan los campos que se mostrarán, se guarda el registro en
            'la base de datos y también los componentes relacionados
Imports System.Data.OleDb
```

```
Public Class GuardarCaso
    Inherits System.Windows.Forms.Form
    Private oDataSet, dsElementos, dsFamilia As DataSet
    Private oDataAdapter, daElementos, daFamilia As OleDbDataAdapter
    Private oDataRow, drElementos, drFamilia As DataRow
    Public i, op As Integer
    Public ar, idFam As String
    Public ancho, largo, prof, dmtro, cap, fac As Double
    Private Sub GuardarCaso_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        'conexion a la base de datos para guardar registros de productos
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.oDataAdapter = New OleDbDataAdapter("select * from BaseDeCasos",
oConexion)
        Me.daElementos = New OleDbDataAdapter("select * from RelComponentes",
oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(oDataAdapter)
        Dim cbElementos As OleDbCommandBuilder = New
OleDbCommandBuilder(daElementos)

        Me.oDataSet = New DataSet
        Me.dsElementos = New DataSet
        oConexion.Open()
        Me.oDataAdapter.Fill(oDataSet, "BaseDeCasos")
        Me.daElementos.Fill(dsElementos, "RelComponentes")
        oConexion.Close()
                Me.i = 0
        productos()
        Me.txtCaso.Focus()
    End Sub

    Private Sub CargarFamilia()
        Dim drFamilia As DataRow
        drFamilia = Me.dsFamilia.Tables("Familia").Rows(Me.i)
        idFam = drFamilia("idFamilia")
        Me.TextBox2.Text = idFam
    End Sub


    Private Sub btnGuardarcom_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnGuardarcom.Click
        Dim drElementos As DataRow
        drElementos = Me.dsElementos.Tables("RelComponentes").NewRow()
        drElementos("Producto") = Me.txtCaso.Text
        drElementos("Componente") = Me.TextBox1.Text

        Me.dsElementos.Tables("RelComponentes").Rows.Add(drElementos)
        Me.daElementos.Update(Me.dsElementos, "RelComponentes")
        MessageBox.Show("Componente guardado")
        Dim i, Ele As Integer
        Ele = Me.cbElementos.SelectedIndex
        i = Me.Label3.Text
        If i = Ele Then
            MessageBox.Show("Ultima componente")
            Me.Close()

            Me.Panel2.Visible = False

        Else
            i += 1
            Me.TextBox1.Clear()


        End If
        Me.Label3.Text = i
```

```
    End Sub

    Private Sub cmbFamilia_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cmbFamilia.SelectedIndexChanged
        Dim fConexion As OleDbConnection
        fConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daFamilia = New OleDbDataAdapter("SELECT * FROM Familia WHERE
Familia LIKE '" & Me.cmbFamilia.SelectedItem & "'", fConexion)
        Dim CBuildFamilia As OleDbCommandBuilder = New
OleDbCommandBuilder(daFamilia)
        Me.dsFamilia = New DataSet
        fConexion.Open()
        Me.daFamilia.Fill(dsFamilia, "Familia")
        fConexion.Close()
        Me.CargarFamilia()
        Me.i = 0
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Me.Close()
    End Sub

    Sub Medidas()
        Select Case op
            Case 1
                fac = 1000
            Case 2
                fac = 10
            Case 3
                fac = 1
            Case 4
                fac = 304.8
            Case 5
                fac = 25.4
        End Select
    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged

        op = Me.ComboBox1.SelectedIndex + 1
        Medidas()
        ancho = Me.txtAncho.Text * fac
    End Sub

    Private Sub ComboBox2_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox2.SelectedIndexChanged
        op = Me.ComboBox2.SelectedIndex + 1
        Medidas()
        largo = Me.txtLargo.Text * fac
    End Sub

    Private Sub ComboBox3_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox3.SelectedIndexChanged
        op = Me.ComboBox3.SelectedIndex + 1
        Medidas()
        prof = Me.txtAlto.Text * fac
    End Sub

    Private Sub ComboBox4_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox4.SelectedIndexChanged
        op = Me.ComboBox4.SelectedIndex + 1
        Medidas()
        dmtro = Me.txtDiametro.Text * fac
    End Sub
```

```
    Sub productos()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM TipoDeProducto",
oConexion)
        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            Me.cbProd.Items.Add(oDataReader("TipoDeProducto"))

        End While
        oDataReader.Close()
        oConexion.Close()
    End Sub
    Private Sub ToolStripMenuItem2_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles toolGuarda.Click
        If Me.txtCaso.Text <> "" Then
            Try
                oDataRow = Me.oDataSet.Tables("BaseDeCasos").NewRow()
                oDataRow("Producto") = Me.txtCaso.Text
                oDataRow("Familia") = Me.cmbFamilia.SelectedItem
                oDataRow("NoElementos") = Me.cbElementos.SelectedItem
                oDataRow("imagen") = Me.txtCaso.Text & ".jpg"
                oDataRow("Geometria") = Me.cbGeometria.SelectedItem
                oDataRow("Ancho") = ancho
                oDataRow("Largo") = largo
                oDataRow("Profundidad") = prof
                oDataRow("Diametro") = dmtro
                oDataRow("Capacidad") = Me.txtCapacidad.Text &
Me.ComboBox5.SelectedItem
                oDataRow("TipoDeLamina") = Me.cbLaCu.SelectedItem
                oDataRow("Calibre") = Me.txtCalibre.Text
                oDataRow("TotalOp") = Me.cbTapa.SelectedItem
                oDataRow("TipoDeProducto") = Me.cbProd.SelectedItem
                Me.oDataSet.Tables("BaseDeCasos").Rows.Add(oDataRow)

                Me.oDataAdapter.Update(Me.oDataSet, "BaseDeCasos")

                MessageBox.Show("Guardado")
                Me.Panel1.Enabled = False
                Me.Panel2.Visible = True


            Catch ex As Exception
                MessageBox.Show("El registro ya existe o faltan campos por
llenar")
                Me.Close()
            End Try
        Else
            MessageBox.Show("Teclea el nombre del componente ")
            Me.Panel2.Visible = False

        End If
        If Me.cbElementos.SelectedIndex = 0 Then
            Me.Close()
            Dim vrt As New RegistarTroqueles
            vrt.Show()

        Else
            Me.Panel2.Enabled = True
        End If
        Me.Label6.Text = Me.cbElementos.SelectedItem
        Me.Label3.Text = "1"
```

```
    End Sub


    Private Sub ToolStripMenuItem3_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles ToolStripMenuItem3.Click
        Try
            Process.Start("C:\Imagenes\Productos")
        Catch ex As Exception
            MessageBox.Show("No se encontro la carpeta correspondiente a
imagenes")
        End Try
    End Sub
    Private Sub ToolCancelar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolCancelar.Click
        Me.Close()
    End Sub
End Class
```

Pantalla: Registrar troqueles

```
          'Se crea el registro que describe las características del troquel
          'junto con las partes que lo componen
Imports System.Data.OleDb
Public Class RegistarTroqueles
    Inherits System.Windows.Forms.Form
    Public i As Integer
    Public dsC, dsF, dsCom, dsGeo, dsTo, dsTro, dsPop, dsPartes As DataSet
    Public daC, daF, daCom, daGeo, daTo, daTro, daPop, daPartes As
OleDbDataAdapter
    Private drTroquel, drPartes As DataRow
    Public codigo(7) As String
    Public codi As String
    Private parte, material, estado As String
    Private cantidad As Integer
    Private Sub RegistarTroquel1_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        Dim oC As OleDbConnection
        oC = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")

        daTro = New OleDbDataAdapter(" select * from Troquel", oC)
        daPartes = New OleDbDataAdapter(" select * from Partes", oC)

        Dim cbTro As OleDbCommandBuilder = New OleDbCommandBuilder(daTro)
        Dim cbPartes As OleDbCommandBuilder = New
OleDbCommandBuilder(daPartes)

        Me.dsTro = New DataSet
        Me.dsPartes = New DataSet

        oC.Open()
        daTro.Fill(dsTro, "Troquel")
        daPartes.Fill(dsPartes, "Partes")
        oC.Close()
        Me.i = 0


        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM BaseDeCasos",
oConexion)
        Dim bComando As New OleDbCommand("SELECT * FROM TipoDeOperacion",
oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
```

```
        Dim bDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        bDataReader = bComando.ExecuteReader()
        ' recorrer filas
        While oDataReader.Read()
            Me.cbProducto.Items.Add(oDataReader("Producto"))

        End While
        oDataReader.Close()
        While bDataReader.Read()
            Me.cbTOper.Items.Add(bDataReader("TipoDeOperacion"))

        End While
        bDataReader.Close()
        oConexion.Close()

    End Sub


    Sub Descripción()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM BaseDeCasos WHERE
Producto LIKE '" & Me.cbProducto.SelectedItem & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            Me.txtFam.Text = oDataReader("Familia")
            Me.txtGeo.Text = oDataReader("Geometria")
            Me.txtTotalOp.Text = oDataReader("TotalOp")
        End While
        oDataReader.Close()
        oConexion.Close()

    End Sub

    Private Sub cbProducto_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles cbProducto.Leave
        llenar()
        Identificadores()
        Me.txtCodigo.Text = codigo(0)
    End Sub

    Private Sub cbProducto_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cbProducto.SelectedIndexChanged
        Descripción()
        Me.cbOperacion.Items.Clear()
        Identificadores()
        Me.txtCodigo.Text = codigo(0)
    End Sub
    Sub llenar()
        Try
            Dim x As Integer
            x = Me.txtTotalOp.Text - 1

            For i = 0 To x Step 1

                Me.cbOperacion.Items.AddRange(New String() {1 + i})

            Next
        Catch ex As Exception
```

```
        End Try


    End Sub
    Sub Identificadores()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM Familia WHERE Familia
LIKE '" & Me.txtFam.Text & "'", oConexion)
        Dim gComando As New OleDbCommand("SELECT * FROM Geometria WHERE
Geometria LIKE '" & Me.txtGeo.Text & "'", oConexion)
        Dim tComando As New OleDbCommand("SELECT * FROM TipoDeOperacion WHERE
TipoDeOperacion LIKE '" & Me.cbTOper.SelectedItem & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        Dim gDataReader As OleDbDataReader
        Dim tDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        gDataReader = gComando.ExecuteReader()
        tDataReader = tComando.ExecuteReader()
        ' recorrer filas
        While oDataReader.Read()
            codigo(0) = oDataReader("IdFamilia")

        End While
        oDataReader.Close()
        While gDataReader.Read()
            codigo(4) = gDataReader("IdGeometria")

        End While
        gDataReader.Close()
        While tDataReader.Read()
            codigo(3) = tDataReader("idOperacion")

        End While
        tDataReader.Close()
        oConexion.Close()
    End Sub

    Private Sub cbTOper_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles cbTOper.Leave
        Identificadores()
        Me.chklPartes.Items.Clear()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM PartesxOp WHERE
idOperacion LIKE '" & codigo(3) & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            Me.chklPartes.Items.Add(oDataReader("Parte"))
        End While
        oDataReader.Close()
        oConexion.Close()
        Me.txtCodigo.Text = codigo(0) & codigo(1) & codigo(2) & codigo(3) &
codigo(4)
```

```
End Sub
Sub IdentificadorOpT()
    Dim total As Integer
    total = Me.txtTotalOp.Text
    If Me.txtTotalOp.Text < 10 Then
        codigo(1) = Me.txtTotalOp.Text
    Else
        Select Case total
            Case 10
                codigo(1) = "A"
            Case 11
                codigo(1) = "B"
            Case 12
                codigo(1) = "C"
            Case 13
                codigo(1) = "D"
            Case 14
                codigo(1) = "E"
            Case 15
                codigo(1) = "F"
            Case 16
                codigo(1) = "G"
            Case 17
                codigo(1) = "H"
            Case 18
                codigo(1) = "J"
            Case 19
                codigo(1) = "K"
            Case 20
                codigo(1) = "L"

        End Select
    End If
End Sub
Sub convertir()
    If Me.cbOperacion.SelectedItem < 10 Then
        codigo(2) = Me.cbOperacion.SelectedItem
    Else
        Dim oper As Integer = Me.cbOperacion.SelectedItem
        Select Case oper
            Case 10
                codigo(2) = "A"
            Case 11
                codigo(2) = "B"
            Case 12
                codigo(2) = "C"
            Case 13
                codigo(2) = "D"
            Case 14
                codigo(2) = "E"
            Case 15
                codigo(2) = "F"
            Case 16
                codigo(2) = "G"
            Case 17
                codigo(2) = "H"
            Case 18
                codigo(2) = "J"
            Case 19
                codigo(2) = "K"
            Case 20
                codigo(2) = "L"

        End Select
    End If
End Sub
```

```
    Private Sub cbOperacion_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
cbOperacion.SelectedIndexChanged
        convertir()
        IdentificadorOpT()
        Me.txtCodigo.Text = codigo(0) & codigo(1) & codigo(2)
    End Sub

    Private Sub cbTipoTroquel_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles cbTipoTroquel.Leave
        If Me.cbTipoTroquel.SelectedIndex = 0 Then
            codigo(5) = "A"
        End If
        If Me.cbTipoTroquel.SelectedIndex = 1 Then
            codigo(5) = "M"
        End If
        Me.txtCodigo.Text = codigo(0) & codigo(1) & codigo(2) & codigo(3) &
codigo(4) & codigo(5)

    End Sub

    Private Sub cbNumero_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cbNumero.SelectedIndexChanged
        codigo(7) = Me.cbNumero.SelectedItem
        Me.txtCodigo.Text = codigo(0) & codigo(1) & codigo(2) & codigo(3) &
codigo(4) & codigo(5) & codigo(6) & codigo(7)
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM Troquel WHERE Codigo
LIKE '" & Me.txtCodigo.Text & "'", oConexion)
        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            MessageBox.Show("Este Código ya existe, elige un numero de
herramienta mayor ")
        End While
        oDataReader.Close()
        oConexion.Close()
    End Sub

    Private Sub cbUso_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cbUso.SelectedIndexChanged
        If Me.cbUso.SelectedIndex = 0 Then
            codigo(6) = 1
        End If
        If Me.cbUso.SelectedIndex = 1 Then
            codigo(6) = 2
        End If
        Me.txtCodigo.Text = codigo(0) & codigo(1) & codigo(2) & codigo(3) &
codigo(4) & codigo(5) & codigo(6)
    End Sub

    Private Sub toolG_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolG.Click
        If codi = 9 Then
            'Dim drTroquel As New DataRow
            drTroquel = Me.dsTro.Tables("Troquel").NewRow()
            drTroquel("Codigo") = Me.txtCodigo.Text
            drTroquel("Familia") = Me.txtFam.Text
            drTroquel("OpDeFamilia") = Me.txtTotalOp.Text
            drTroquel("Operacion") = Me.cbOperacion.SelectedIndex + 1
            drTroquel("TipoDeOperacion") = Me.cbTOper.SelectedItem
            drTroquel("Geometria") = Me.txtGeo.Text
```

```
            drTroquel("TipoDeTroquel") = Me.cbTipoTroquel.SelectedItem
            drTroquel("Producto") = Me.cbProducto.SelectedItem
            drTroquel("Ubicacion") = Me.cbUbic.SelectedItem
            drTroquel("Localizacion") = Me.txtLocalizacion.Text
            drTroquel("ConstanciaDeUso") = Me.cbUso.SelectedItem
            drTroquel("Observaciones") = Me.txtObserva.Text
            drTroquel("Imagen") = Me.txtCodigo.Text & "." & "jpg"

            Me.dsTro.Tables("Troquel").Rows.Add(drTroquel)


            Try
                Me.daTro.Update(Me.dsTro, "Troquel")
                MessageBox.Show("Registro guardado")
                Me.Panel1.Enabled = False
                Me.Panel4.Enabled = True
            Catch ex As OleDbException
                MessageBox.Show("Este código ya fue guardado anteriormente")

            End Try

        Else
            MessageBox.Show("El código no esta completo, faltan campos por
llenar")
        End If
    End Sub

    Private Sub txtCodigo_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles txtCodigo.TextChanged
        codi = CStr(Len(txtCodigo.Text))
    End Sub

    Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles CheckBox1.CheckedChanged
        Me.chklPartes.Items.Clear()
        If Me.CheckBox1.Checked = True Then
            Dim oConexion As New OleDbConnection()
            oConexion.ConnectionString =
"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=BDGZ.mdb;"
            ' crear comando
            Dim oComando As New OleDbCommand("SELECT * FROM PartesxOp WHERE
idOperacion LIKE '" & "G" & "'", oConexion)

            ' crear DataReader
            Dim oDataReader As OleDbDataReader
            oConexion.Open()
            oDataReader = oComando.ExecuteReader() ' obtener DataReader
            ' recorrer filas
            While oDataReader.Read()
                Me.chklPartes.Items.Add(oDataReader("Parte"))
            End While
            oDataReader.Close()
            oConexion.Close()
        Else
            Me.chklPartes.Items.Clear()
        End If

    End Sub

    Private Sub ToolImG_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolImG.Click
        Try
            Process.Start("C:\Imagenes")
        Catch ex As Exception
            MessageBox.Show("No se encontro la carpeta correspondiente a
imagenes")
        End Try
    End Sub
```

```
    Private Sub chklPartes_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles chklPartes.SelectedIndexChanged
        If Me.chklPartes.CheckOnClick = True Then
            parte = Me.chklPartes.SelectedItem.ToString
            Me.txtPartes.Text = parte

            Try
                material = InputBox("Tipo de material", "", "desconocido")
                cantidad = InputBox("Cantidad", "Cantidad", "1")
                estado = InputBox("Estado de la parte", "Estado de la parte",
"Bueno")
            Catch ex As Exception

            End Try


        Else

        End If
'Guarda las características de cada parte del troquel si existe ya registrado
'un troquel
        If MessageBox.Show("Deseas guardar este registro", "Guardar",
MessageBoxButtons.YesNo, MessageBoxIcon.Question) =
Windows.Forms.DialogResult.Yes Then

            Try

                drPartes = Me.dsPartes.Tables("Partes").NewRow()
                drPartes("Codigo") = Me.txtCodigo.Text
                drPartes("Parte") = Me.txtPartes.Text
                drPartes("Material") = material
                drPartes("Cantidad") = cantidad
                drPartes("EstadoFisico") = estado
                Me.dsPartes.Tables("Partes").Rows.Add(drPartes)

                Me.daPartes.Update(Me.dsPartes, "Partes")
            Catch ex As Exception
                MessageBox.Show("Necesitas crear un código")
                Dim x As Integer
                x = Me.chklPartes.SelectedIndex
                MessageBox.Show("no se guardaron los datos")
                Me.chklPartes.SetItemChecked(x, False)
            End Try
        Else
            Dim x As Integer
            x = Me.chklPartes.SelectedIndex
            MessageBox.Show("no se guardaron los datos")
            Me.chklPartes.SetItemChecked(x, False)
        End If


    End Sub

    Private Sub cbTOper_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cbTOper.SelectedIndexChanged
        Identificadores()
        Me.txtCodigo.Text = codigo(0) & codigo(1) & codigo(2) & codigo(3)
    End Sub

    Private Sub txtLocalizacion_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles txtLocalizacion.TextChanged
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM Troquel WHEre
Localizacion LIKE '" & Me.txtLocalizacion.Text & "'", oConexion)
```

```
        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            MessageBox.Show("Este espacio esta ocupado")
        End While
        oDataReader.Close()
        oConexion.Close()
    End Sub


    Private Sub ToopCancelar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToopCancelar.Click
        Me.Close()
    End Sub
End Class
```

Pantalla: Modificar registros

```
'Modificar los registros ya guardados en la base de datos, sólo los campos que
'no intervienen en la creación del código
Imports System.Data.OleDb
Public Class Modificar
    Inherits System.Windows.Forms.Form
    Private daTroquel, daPartes, oDataAdapter, daT As OleDbDataAdapter
    Private dsTroquel, dsPartes, oDataSet, dsT As DataSet
    Private drT, drTroquel As DataRow
    Private oConexion As OleDbConnection
    Private temp, valor As String
    Private ipos, rowy, colx As Integer
    Dim pos As Integer
    Dim ubi As String
    Dim loc As String
    Dim uso As String
    Dim obse As String


    Dim troq(4) As String


    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daPartes = New OleDbDataAdapter("SELECT * FROM Partes", oConexion)
        Me.daTroquel = New OleDbDataAdapter("SELECT * FROM Troquel",
oConexion)
        Dim parteCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daPartes)
        Dim troquelpCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daTroquel)
        Me.dsPartes = New DataSet
        Me.dsTroquel = New DataSet
        oConexion.Open()
        Me.daPartes.Fill(dsPartes, "Partes")
        Me.daTroquel.Fill(dsTroquel, "Troquel")
        oConexion.Close()
        Me.ipos = 0
        Me.DataGridView1.DataSource = dsTroquel
        Me.DataGridView1.DataMember = "Troquel"
        ' asignar al datagrid detalles la relación
        ' que acabamos de crear por código
        Me.DataGridView2.DataSource = dsTroquel
        Me.ipos = 0
        Me.DataGridView1.Columns(0).ReadOnly = True
```

```
        Me.DataGridView1.Columns(1).ReadOnly = True
        Me.DataGridView1.Columns(1).Width = 200
        Me.DataGridView1.Columns(2).ReadOnly = True
        Me.DataGridView1.Columns(2).Width = 20
        Me.DataGridView1.Columns(3).ReadOnly = True
        Me.DataGridView1.Columns(3).Width = 20
        Me.DataGridView1.Columns(4).ReadOnly = True
        Me.DataGridView1.Columns(5).ReadOnly = True
        Me.DataGridView1.Columns(6).ReadOnly = True

        Me.DataGridView1.Columns(12).ReadOnly = True
        Me.DataGridView1.Columns(11).Visible = False

    End Sub

    Private Sub btnGrabar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnGrabar.Click
        Try

            Me.daTroquel.Update(dsTroquel, "Troquel")
            Me.daPartes.Update(dsTroquel, "Partes")

        Catch ex As OleDbException
            MessageBox.Show("No puedes crear registros en esta pantalla")
        End Try


    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        If MessageBox.Show(" Al eliminar este registro también se eliminan sus
partes, ¿Está seguro de ELIMINAR este registro?", "Eliminar Registro",
MessageBoxButtons.YesNo, MessageBoxIcon.Question) =
Windows.Forms.DialogResult.Yes Then
            Try

            Catch ex As Exception
                MessageBox.Show("Selecciona todo el renglón")
            End Try
        End If


    End Sub

    Private Sub DataGridView1_CellContentClick(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView1.CellContentClick

    End Sub

    Private Sub DataGridView1_CellMouseClick(ByVal sender As Object, ByVal e
As System.Windows.Forms.DataGridViewCellMouseEventArgs) Handles
DataGridView1.CellMouseClick
        Try


            Me.TextBox1.Text = Me.DataGridView1.CurrentCellAddress.Y

            rowy = Me.DataGridView1.CurrentCellAddress.Y
            Dim temp As String = dsTroquel.Tables("Troquel").Rows(rowy)(11)
            PictureBox1.Image = Image.FromFile("C:\Imagenes\" & temp)
        Catch ex As Exception
            PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        End Try
    End Sub
```

```
    Private Sub toolGuardar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolGuardar.Click
        Try

            Me.daTroquel.Update(dsTroquel, "Troquel")
            MessageBox.Show("Modificación realizada")
        Catch ex As OleDbException '
            MessageBox.Show("No puedes crear registros en esta pantalla")
        End Try

    End Sub

    Private Sub ToolEliminar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolEliminar.Click
        If MessageBox.Show(" Al eliminar este registro también se eliminan sus
partes, ¿Está seguro de ELIMINAR este registro?", "Eliminar Registro",
MessageBoxButtons.YesNo, MessageBoxIcon.Question) =
Windows.Forms.DialogResult.Yes Then
            Try
                Dim obDataRow As DataRow
                obDataRow = Me.dsTroquel.Tables("Troquel").Rows(rowy)
                obDataRow.Delete()
                Dim oTablaBorrados As DataTable
                oTablaBorrados = _

Me.dsTroquel.Tables("Troquel").GetChanges(DataRowState.Deleted)
                Me.daTroquel.Update(oTablaBorrados)
                Me.dsTroquel.Tables("Troquel").AcceptChanges()
                Me.btnpri.PerformClick()
            Catch ex As Exception
                MessageBox.Show("Selecciona todo el renglón")
            End Try
        End If

    End Sub

    Private Sub ToolStripButton1_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ToolStripButton1.Click
        Me.Close()
    End Sub


    Private Sub toolGuardar_MouseDown(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles toolGuardar.MouseDown
        Me.TextBox2.Focus()
    End Sub


    Private Sub ToolStripButton3_MouseDown(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs)
    End Sub

    Private Sub DataGridView2_CellContentClick(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView2.CellContentClick
    End Sub


    Private Sub DataGridView1_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles DataGridView1.MouseClick
        Dim codigo As String
        codigo = Me.DataGridView1.CurrentCell.Value
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
```

```
        Me.daPartes = New OleDbDataAdapter("SELECT * FROM Partes where codigo
like '" & codigo & "'", oConexion)

        Dim parteCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daPartes)
        Me.dsPartes = New DataSet
        oConexion.Open()
        Me.daPartes.Fill(dsPartes, "Partes")
        oConexion.Close()
        Me.ipos = 0
        Me.DataGridView2.DataSource = dsPartes
        Me.DataGridView2.DataMember = "Partes"
    End Sub
End Class
```

Pantalla: Buscar

```
'Se muestrán varias tablas y su relaciones, se muestrán varios campos de los
'troqueles
Imports System.Data.OleDb
Public Class Buscar
    Inherits System.Windows.Forms.Form
    Private dsFamilia, dsGeometria, dsOperacion, dsPartes, dsTroquel, dsDif1,
dsDif2, dsBDC As DataSet
    Private daFamilia, daGeometria, daOperacion, daPartes, daTroquel, daDif1,
daDif2, daBDC As OleDbDataAdapter
    Private drFamilia, drGeometria, drOperacion, drPartes, drTroquel, drDif1,
drDif2, drBDC As DataRow
    Private oConexion As OleDbConnection
    Public iResultado As Integer
    Public ipos, rowy As Integer
    Private temp As String
    Private parte, material, cantidad, estado, dimen, idFam, idGeo, idOp, cod
As String
    Private Codigo(7) As String

    Private Sub DataGrid1_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles DataGrid1.MouseClick
        Try
            rowy = Me.DataGrid1.CurrentCell.RowNumber
            Dim temp As String = Me.DataGrid1.Item(rowy, 11)
            PictureBox1.Image = Image.FromFile("C:\Imagenes\" & temp)
        Catch ex As Exception
            PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        End Try
    End Sub

    Private Sub DataGridView1_MouseClick1(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles DataGridView1.MouseClick
        Try
            rowy = Me.DataGridView1.CurrentCellAddress.Y
            Dim temp As String = dsTroquel.Tables("Troquel").Rows(rowy)(13)
            PictureBox2.Image = Image.FromFile("C:\Imagenes\" & temp)
        Catch ex As Exception
            PictureBox2.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        End Try
    End Sub

    Private Sub toolFamilia_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolFamilia.Click
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daFamilia = New OleDbDataAdapter("SELECT * FROM Familia",
oConexion)
        Me.daTroquel = New OleDbDataAdapter("SELECT * FROM Troquel",
oConexion)
        Dim parteCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daPartes)
```

```
        Dim troquelpCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daTroquel)
        Me.dsFamilia = New DataSet
        Me.dsTroquel = New DataSet
        oConexion.Open()
        Me.daFamilia.Fill(dsFamilia, "Familia")
        Me.daTroquel.Fill(dsFamilia, "Troquel")
        oConexion.Close()
        Me.ipos = 0
        dsFamilia.Relations.Add("Familia_Troquel", _
               dsFamilia.Tables("Familia").Columns("Familia"), _
               dsFamilia.Tables("Troquel").Columns("Familia"))
        ' asignar al datagrid maestro la tabla Customers
        Me.DataGridView1.DataSource = dsFamilia
        Me.DataGridView1.DataMember = "Familia"
        ' asignar al datagrid detalles la relación
        ' que acabamos de crear por código
        Me.DataGrid1.DataSource = dsFamilia
        Me.DataGrid1.DataMember = "Familia.Familia_Troquel"
        Me.PictureBox2.Visible = False
        Me.PictureBox1.Visible = True
        PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        PictureBox2.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")

        Me.DataGridView1.Columns(0).Width = 20
        Me.DataGridView1.Columns(1).Width = 150



    End Sub

    Private Sub toolGeo_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolGeo.Click
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daGeometria = New OleDbDataAdapter("SELECT * FROM Geometria",
oConexion)
        Me.daTroquel = New OleDbDataAdapter("SELECT * FROM Troquel",
oConexion)
        Dim parteCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daPartes)
        Dim troquelpCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daTroquel)
        Me.dsGeometria = New DataSet
        Me.dsTroquel = New DataSet
        oConexion.Open()
        Me.daGeometria.Fill(dsGeometria, "Geometria")
        Me.daTroquel.Fill(dsGeometria, "Troquel")
        oConexion.Close()
        'Me.Cargar()
        Me.ipos = 0
        dsGeometria.Relations.Add("Geometria_Troquel", _
               dsGeometria.Tables("Geometria").Columns("Geometria"), _
               dsGeometria.Tables("Troquel").Columns("Geometria"))
        ' asignar al datagrid maestro la tabla Customers
        Me.DataGridView1.DataSource = dsGeometria
        Me.DataGridView1.DataMember = "Geometria"
        ' asignar al datagrid detalles la relación
        ' que acabamos de crear por código
        Me.DataGrid1.DataSource = dsGeometria
        Me.DataGrid1.DataMember = "Geometria.Geometria_Troquel"
        Me.PictureBox2.Visible = False
        Me.PictureBox1.Visible = True
        PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        PictureBox2.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        Me.DataGridView1.Columns(0).Width = 20
        Me.DataGridView1.Columns(1).Width = 150
```

```
    End Sub

    Private Sub toolOper_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolOper.Click
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daOperacion = New OleDbDataAdapter("SELECT * FROM TipoDeOperacion",
oConexion)
        Me.daTroquel = New OleDbDataAdapter("SELECT * FROM Troquel",
oConexion)
        Dim parteCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daPartes)
        Dim troquelpCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daTroquel)
        Me.dsOperacion = New DataSet
        Me.dsTroquel = New DataSet
        oConexion.Open()
        Me.daOperacion.Fill(dsOperacion, "TipoDeOperacion")
        Me.daTroquel.Fill(dsOperacion, "Troquel")
        oConexion.Close()
        Me.ipos = 0
        dsOperacion.Relations.Add("TipoDeOperacion_Troquel", _

dsOperacion.Tables("TipoDeOperacion").Columns("TipoDeOperacion"), _
             dsOperacion.Tables("Troquel").Columns("TipoDeOperacion"))
        Me.DataGridView1.DataSource = dsOperacion
        Me.DataGridView1.DataMember = "TipoDeOperacion"
        Me.DataGrid1.DataSource = dsOperacion
        Me.DataGrid1.DataMember = "TipoDeOperacion.TipoDeOperacion_Troquel"
        Me.PictureBox2.Visible = False
        Me.PictureBox1.Visible = True
        PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        PictureBox2.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        Me.DataGridView1.Columns(0).Width = 20
        Me.DataGridView1.Columns(1).Width = 150
    End Sub

    Private Sub toolTroquel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolTroquel.Click
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daPartes = New OleDbDataAdapter("SELECT * FROM Partes", oConexion)
        Me.daTroquel = New OleDbDataAdapter("SELECT * FROM Troquel",
oConexion)
        Dim parteCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daPartes)
        Dim troquelpCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daTroquel)
        Me.dsPartes = New DataSet
        Me.dsTroquel = New DataSet
        oConexion.Open()
        Me.daPartes.Fill(dsTroquel, "Partes")
        Me.daTroquel.Fill(dsTroquel, "Troquel")
        oConexion.Close()
        Me.ipos = 0
        dsTroquel.Relations.Add("Troquel_Partes", _
               dsTroquel.Tables("Troquel").Columns("Codigo"), _
               dsTroquel.Tables("Partes").Columns("codigo"))
        Me.DataGridView1.DataSource = dsTroquel
        Me.DataGridView1.DataMember = "Troquel"
        Me.DataGrid1.DataSource = dsTroquel
        Me.DataGrid1.DataMember = "Troquel.Troquel_Partes"
        Me.PictureBox2.Visible = True
        Me.PictureBox1.Visible = False
        PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        PictureBox2.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
```

```
    End Sub

    Private Sub toolprod_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolprod.Click
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daBDC = New OleDbDataAdapter("SELECT * FROM BaseDeCasos",
oConexion)
        Me.daTroquel = New OleDbDataAdapter("SELECT * FROM Troquel",
oConexion)
        Dim BDCCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daBDC)
        Dim troquelpCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daTroquel)
        Me.dsBDC = New DataSet
        Me.dsTroquel = New DataSet
        oConexion.Open()
        Me.daBDC.Fill(dsBDC, "BaseDeCasos")
        Me.daTroquel.Fill(dsBDC, "Troquel")
        oConexion.Close()
        Me.ipos = 0
        dsBDC.Relations.Add("BaseDeCasos_Troquel", _
               dsBDC.Tables("BaseDeCasos").Columns("Producto"), _
               dsBDC.Tables("Troquel").Columns("Producto"))
        Me.DataGridView1.DataSource = dsBDC
        Me.DataGridView1.DataMember = "BaseDeCasos"
        Me.DataGrid1.DataSource = dsBDC
        Me.DataGrid1.DataMember = "BaseDeCasos.BaseDeCasos_Troquel"
        Me.PictureBox2.Visible = False
        Me.PictureBox1.Visible = True
        PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        PictureBox2.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
    End Sub

    Private Sub toolBuscar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolBuscar.Click
        Dim v As New BusquedaLibre
        v.Show()
    End Sub


    Private Sub ToolStripButton1_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles ToolStripButton1.Click
        Me.Close()
    End Sub
End Class
```

Pantalla: Búsqueda libre

```
'Se trabaja sobre la tabla Troqueles, eligiendo un campo e introduciendo un
'dato que será buscado en esta tabla.
Imports System.Data.OleDb
Public Class BusquedaLibre
    Inherits System.Windows.Forms.Form
    Private daTroquel, daPartes As OleDbDataAdapter
    Private dsTroquel, dsPartes As DataSet
    Private oConexion As OleDbConnection
    Private temp As String
    Private ipos, rowy As Integer
   Private Sub BusquedaLibre_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daTroquel = New OleDbDataAdapter("SELECT * FROM Troquel",
oConexion)
```

```
        Dim troquelpCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(daTroquel)
        Me.dsTroquel = New DataSet
        oConexion.Open()
        Me.daTroquel.Fill(dsTroquel, "Troquel")
        oConexion.Close()
        Me.ipos = 0
        Me.DataGrid1.DataSource = dsTroquel
        Me.DataGrid1.DataMember = "Troquel"

    End Sub
'El evento de este botón realiza un filtro de los datos que contengan el
'campo y el dato introducidos para desplegar la información
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim oDataView As New DataView()
        oDataView.Table = dsTroquel.Tables("Troquel")
        Dim campo As String
        campo = Me.ComboBox1.SelectedItem
        If Me.ComboBox1.SelectedIndex = 0 Then
            oDataView.RowFilter = "Codigo='" & Me.TextBox1.Text & "' "

            Me.DataGrid1.CaptionText = "Busqueda por: " & Me.ComboBox1.Text
            Me.DataGrid1.DataSource = oDataView

        End If
        If Me.ComboBox1.SelectedIndex = 1 Then
            Try
                oDataView.RowFilter = "OpDeFamilia='" & Me.TextBox1.Text & "'
"

                Me.DataGrid1.CaptionText = "Busqueda por: " &
Me.ComboBox1.Text
                Me.DataGrid1.DataSource = oDataView
            Catch ex As Exception
                MessageBox.Show("El tipo de dato debe ser numerico")
            End Try

        End If
        If Me.ComboBox1.SelectedIndex = 2 Then
            Try
                oDataView.RowFilter = "Operacion='" & Me.TextBox1.Text & "' "

                Me.DataGrid1.CaptionText = "Busqueda por: " &
Me.ComboBox1.Text
                Me.DataGrid1.DataSource = oDataView
            Catch ex As Exception
                MessageBox.Show("El tipo de dato debe ser numerico")
            End Try

        End If
        If Me.ComboBox1.SelectedIndex = 4 Then
            oDataView.RowFilter = "Ubicacion='" & Me.TextBox1.Text & "' "

            Me.DataGrid1.CaptionText = "Busqueda por: " & Me.ComboBox1.Text
            Me.DataGrid1.DataSource = oDataView
        End If
        If Me.ComboBox1.SelectedIndex = 5 Then
            oDataView.RowFilter = "Localizacion='" & Me.TextBox1.Text & "' "

            Me.DataGrid1.CaptionText = "Busqueda por: " & Me.ComboBox1.Text
            Me.DataGrid1.DataSource = oDataView
        End If
        If Me.ComboBox1.SelectedIndex = 6 Then
           oDataView.RowFilter = "ConstanciaDeUso='" & Me.TextBox1.Text & "' "
       Me.DataGrid1.CaptionText = "Busqueda por: " & Me.ComboBox1.Text
            Me.DataGrid1.DataSource = oDataView
        End If
```

```
    End Sub

    Private Sub DataGrid1_MouseClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles DataGrid1.MouseClick
        Try
            rowy = Me.DataGrid1.CurrentCell.RowNumber
            Dim temp As String = Me.DataGrid1.Item(rowy, 11)

            PictureBox1.Image = Image.FromFile("C:\Imagenes\" & temp)
        Catch ex As Exception
            PictureBox1.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        End Try

    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged
        Me.TextBox1.Clear()
    End Sub
End Class
```

Pantalla: Buscar similitudes

```
'Se introduce el método de razonamiento basado en casos para encontrar las
'similitudes entre los datos introducidos por el usuario y la base de casos

Imports System.Data.OleDb
Public Class rbcN
    Inherits System.Windows.Forms.Form
    Private oDataSet, dsElementos, dsFamilia, dsCaso, dsTroquel As DataSet
    Private oDataAdapter, daElementos, daFamilia, daCaso, daTroquel As
OleDbDataAdapter
    Private oDataRow, drElementos, drFamilia As DataRow

    Public ar, idFam, comp, tro As String
    Public suma, suma1, suma2, op, rowy As Integer
    Public i, a, b, c, d, wi As Integer
    Public sim As Double
    Public Atributos(11) As Integer
    Public Elementos(11) As String
    Public Ejemplos(11) As String
    Public resultado(11) As Double
    Public Pesos(11) As Integer
    Public Caso(11) As String
    Public ancho, largo, prof, dmtro, cap, fac As Double

    Private Sub GuardarCaso_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.oDataAdapter = New OleDbDataAdapter("select * from BaseDeCasos",
oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(oDataAdapter)

        Me.oDataSet = New DataSet
        oConexion.Open()
        Me.oDataAdapter.Fill(oDataSet, "BaseDeCasos")
        oConexion.Close()
        Me.i = 0
        Cargar()
        Me.DataGridView1.Columns(0).Width = 140
        Me.DataGridView1.Columns(1).Width = 50


    End Sub
```

```
    Private Sub CargarFamilia()
        Try
            Dim drFamilia As DataRow
            drFamilia = Me.dsFamilia.Tables("Familia").Rows(Me.i)
            idFam = drFamilia("idFamilia")
            Me.TextBox2.Text = idFam
        Catch ex As Exception
            MessageBox.Show("Desea ingresar un nuevo caso ")
        End Try

    End Sub
    Private Sub cmbFamilia_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles cmbFamilia.SelectedIndexChanged
        Dim fConexion As OleDbConnection
        fConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        Me.daFamilia = New OleDbDataAdapter("SELECT * FROM Familia WHERE
Familia LIKE '" & Me.cmbFamilia.SelectedItem & "'", fConexion)
        Dim CBuildFamilia As OleDbCommandBuilder = New
OleDbCommandBuilder(daFamilia)
        Me.dsFamilia = New DataSet
        fConexion.Open()
        Me.daFamilia.Fill(dsFamilia, "Familia")
        fConexion.Close()
        Me.CargarFamilia()
        'Me.Cargar()
        Me.i = 0
        productos()
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Me.Close()
    End Sub
    Sub Cargar()

        oDataRow = Me.oDataSet.Tables("BaseDeCasos").Rows(Me.i)
        Me.LinkLabel1.Text = oDataRow("Producto")
        Ejemplos(0) = oDataRow("Familia")
        Ejemplos(1) = oDataRow("NoElementos")
        Ejemplos(2) = oDataRow("Geometria")
        Ejemplos(3) = oDataRow("TotalOp")

        Ejemplos(4) = oDataRow("TipoDeLamina")
        Ejemplos(5) = oDataRow("Calibre")
        Ejemplos(6) = oDataRow("TipoDeProducto")

        Ejemplos(7) = oDataRow("Ancho")
        Ejemplos(8) = oDataRow("Largo")
        Ejemplos(9) = oDataRow("Profundidad")
        Ejemplos(10) = oDataRow("Diametro")
        Ejemplos(11) = oDataRow("Capacidad")
    End Sub
    Sub Comparar()

        For c = 0 To 11 Step 1
            Me.comparacion()
'Pesos asignados a cada atributo del caso
            suma += resultado(c)
            Pesos(0) = 5
            Pesos(1) = 1
            Pesos(2) = 2
            Pesos(3) = 1
            Pesos(4) = 1
            Pesos(5) = 1
            Pesos(6) = 1
            Pesos(7) = 2
            Pesos(8) = 2
```

```
            Pesos(9) = 2
            Pesos(10) = 2
            Pesos(11) = 2
            Atributos(c) = resultado(c) * Pesos(c)

            wi = Pesos(0) + Pesos(1) + Pesos(2) + Pesos(3) + Pesos(4) +
Pesos(5) + Pesos(6) + Pesos(7) + Pesos(8) + Pesos(9) + Pesos(10) + Pesos(11)
'aplicación de la técnica del vecino más cercano
            suma1 += Atributos(c)
            sim = suma1 / wi
            Me.TextBox2.Text = sim
            Me.TextBox3.Text = (1 - sim) * 100 'porcentaje de similitud


        Next

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Me.ListBox4.Items.Clear()
        For b = 0 To 11 Step 1
             'Asignación de indices del nuevo caso
            Elementos(0) = Me.cmbFamilia.SelectedItem
            Elementos(1) = Me.cbElementos.SelectedItem
            Elementos(2) = Me.cbGeometria.SelectedItem
            Elementos(3) = Me.cbTapa.SelectedItem
           Elementos(4) = Me.cbLaCu.SelectedItem
            Elementos(5) = Me.txtCalibre.Text
            Elementos(6) = Me.cbProd.SelectedItem
            Elementos(7) = ancho
            Elementos(8) = largo
            Elementos(9) = prof
            Elementos(10) = dmtro
            Elementos(11) = Me.txtCapacidad.Text & Me.ComboBox5.SelectedItem
            Try
                Me.ListBox4.Items.Add(Elementos(b))
            Catch ex As Exception
                MessageBox.Show("Faltan elementos que mostrar")
            End Try

        Next
        Me.Panel1.Enabled = False
    End Sub

    Sub comparacion()   'Reglas de comparación entre los atributos
        If Ejemplos(0) = Elementos(0) Then
            resultado(0) = 0
        Else
            resultado(0) = 1
        End If
        If Ejemplos(1) = Elementos(1) Then
            resultado(1) = 0
        Else
            resultado(1) = 1
        End If
        If Ejemplos(2) = Elementos(2) Then
            resultado(2) = 0
        Else
            resultado(2) = 1
        End If
        If Ejemplos(3) = Elementos(3) Then
            resultado(3) = 0
        Else
            resultado(3) = 1
        End If
        If Ejemplos(4) = Elementos(4) Then
            resultado(4) = 0
```

```
    Else
        resultado(4) = 1
    End If
    If Ejemplos(5) = Elementos(5) Then
        resultado(5) = 0
    Else
        resultado(5) = 1
    End If
    If Ejemplos(6) = Elementos(6) Then
        resultado(6) = 0
    Else
        resultado(6) = 1
    End If
    If Ejemplos(7) = Elementos(7) Then
        resultado(7) = 0
    ElseIf Elementos(7) > (Ejemplos(7) + 5) Then
        resultado(7) = 1
    ElseIf Ejemplos(7) < (Elementos(7) + 5) Then
        resultado(7) = 0.5
    ElseIf (Elementos(7) - 5) > Ejemplos(7) Then
        resultado(7) = 1
    End If
    If Ejemplos(8) = Elementos(8) Then
        resultado(8) = 0
    ElseIf Elementos(8) > (Ejemplos(8) + 5) Then
        resultado(8) = 1
    ElseIf Ejemplos(8) < (Elementos(8) + 5) Then
        resultado(8) = 0.5
    ElseIf (Elementos(8) - 5) > Ejemplos(8) Then
        resultado(8) = 1
    End If
    If Ejemplos(9) = Elementos(9) Then
        resultado(9) = 0
    ElseIf Elementos(9) > (Ejemplos(9) + 5) Then
        resultado(9) = 1
    ElseIf Ejemplos(9) < (Elementos(9) + 5) Then
        resultado(9) = 0.5
    ElseIf (Elementos(9) - 5) > Ejemplos(9) Then
        resultado(9) = 1
    End If
    If Ejemplos(10) = Elementos(10) Then
        resultado(10) = 0
    ElseIf Elementos(10) > (Ejemplos(10) + 5) Then
        resultado(10) = 1
    ElseIf Ejemplos(10) < (Elementos(10) + 5) Then
        resultado(10) = 0.5
    ElseIf (Elementos(9) - 5) > Ejemplos(10) Then
        resultado(10) = 1
    End If
    If Ejemplos(11) = Elementos(11) Then
        resultado(11) = 0
    Else
        resultado(11) = 1
    End If


End Sub


Sub Medidas()
    Select Case op
        Case 1
            fac = 1000
        Case 2
            fac = 10
        Case 3
            fac = 1
        Case 4
```

```
                fac = 304.8
            Case 5
                fac = 25.4
        End Select
    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged

        op = Me.ComboBox1.SelectedIndex + 1
        Medidas()
        ancho = Me.txtAncho.Text * fac
    End Sub

    Private Sub ComboBox2_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox2.SelectedIndexChanged
        op = Me.ComboBox2.SelectedIndex + 1
        Medidas()
        largo = Me.txtLargo.Text * fac
    End Sub

    Private Sub ComboBox3_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox3.SelectedIndexChanged
        op = Me.ComboBox3.SelectedIndex + 1
        Medidas()
        prof = Me.txtAlto.Text * fac
    End Sub

    Private Sub ComboBox4_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox4.SelectedIndexChanged
        op = Me.ComboBox4.SelectedIndex + 1
        Medidas()
        dmtro = Me.txtDiametro.Text * fac
    End Sub

    Private Sub btnGuardar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnGuardar.Click

        If Me.i = _
                            (Me.oDataSet.Tables("BaseDeCasos").Rows.Count - 1)
Then
            MessageBox.Show("Último registro")

        Else
            For Me.i = 0 To Me.oDataSet.Tables("BaseDeCasos").Rows.Count - 1
Step 1
                suma = 0
                suma1 = 0
                Me.Cargar()
                Dim oDataRow As DataRow
                oDataRow = oDataSet.Tables("BaseDeCasos").Rows(i)
                Me.Comparar()
                Me.DataGridView1.Rows.Add(Me.LinkLabel1.Text, " " &
Me.TextBox3.Text) 'Desplega los casos similares


            Next
        End If

        Me.DataGridView1.Sort(Me.DataGridView1.Columns(1),
System.ComponentModel.ListSortDirection.Descending) 'ordena los casos
    End Sub

    Private Sub DataGridView1_CellContentClick(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.DataGridViewCellEventArgs) Handles
DataGridView1.CellContentClick

        comp = Me.DataGridView1.CurrentCell.Value
```

```
Dim ven As New Resultado
ven.prod = Me.DataGridView1.CurrentCell.Value

ven.ListBox1.Items.Clear()
ven.ListBox2.Items.Clear()
ven.ListBox3.Items.Clear()

ven.TextBox1.Text = "Nombre del componente: " & comp
ven.Descripción()
ven.llenarCheck()
For c = 0 To 11 Step 1
    Descripción()
    If Caso(c) = Elementos(c) Then
        ven.sim(c) = "Si"
    Else
        ven.sim(c) = "No"

    End If
    comparacion2()
    If Caso(0) = Elementos(0) Then
        ven.sim(0) = "Si"
    Else
        ven.sim(0) = "No"
    End If
    If Caso(1) = Elementos(1) Then
        ven.sim(1) = "Si"
    Else
        ven.sim(1) = "No"
    End If
    If Caso(2) = Elementos(2) Then
        ven.sim(2) = "Si"
    Else
        ven.sim(2) = "No"
    End If
    If Caso(3) = Elementos(3) Then
        ven.sim(3) = "Si"
    Else
        ven.sim(3) = "No"
    End If
    If Caso(4) = Elementos(4) Then
        ven.sim(4) = "Si"
    Else
        ven.sim(4) = "No"
    End If
    If Caso(5) = Elementos(5) Then
        ven.sim(5) = "Si"
    Else
        ven.sim(5) = "No"
    End If
    If Caso(6) = Elementos(6) Then
        ven.sim(6) = "Si"
    Else
        ven.sim(6) = "No"
    End If
    If Caso(7) = Elementos(7) Then
        ven.sim(7) = "Si"
    ElseIf Elementos(7) > (Caso(7) + 5) Then
        ven.sim(7) = "No"
    ElseIf Caso(7) < (Elementos(7) + 5) Then
        ven.sim(7) = "aproximado"
    ElseIf (Elementos(7) - 5) > Caso(7) Then
        ven.sim(7) = "No"
    End If
    If Caso(8) = Elementos(8) Then
        ven.sim(8) = "Si"
    ElseIf Elementos(8) > (Caso(8) + 5) Then
        ven.sim(8) = "No"
```

```
            ElseIf Caso(8) < (Elementos(8) + 5) Then
                ven.sim(8) = "aproximado"
            ElseIf (Elementos(8) - 5) > Caso(8) Then
                ven.sim(8) = "No"
            End If
            If Caso(9) = Elementos(9) Then
                ven.sim(9) = "Si"
            ElseIf Elementos(9) > (Caso(9) + 5) Then
                ven.sim(9) = "No"
            ElseIf Caso(9) < (Elementos(9) + 5) Then
                ven.sim(9) = "aproximado"
            ElseIf (Elementos(9) - 5) > Caso(9) Then
                ven.sim(9) = "No"
            End If
            If Caso(10) = Elementos(10) Then
                ven.sim(10) = "Si"
            ElseIf Elementos(10) > (Caso(10) + 5) Then
                ven.sim(10) = "No"
            ElseIf Caso(10) < (Elementos(10) + 5) Then
                ven.sim(10) = "aproximado"
            ElseIf (Elementos(9) - 5) > Caso(10) Then
                ven.sim(10) = "No"
            End If
            If Caso(11) = Elementos(11) Then
                ven.sim(11) = "Si"
            Else
                ven.sim(11) = "No"
            End If

            ven.ListBox3.Items.Add(ven.sim(c))
        Next

        Try
            Dim temp As String = comp
            ven.PictureBox1.Image = Image.FromFile("C:\Imagenes\Productos\" &
temp & ".jpg")
        Catch ex As Exception
            ven.PictureBox1.Image =
Image.FromFile("C:\Imagenes\Productos\SinImagen.jpg")
        End Try
        ven.Relacion()

        ven.Show()
    End Sub

    Sub Descripción()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        Dim oComando As New OleDbCommand("SELECT * FROM BaseDeCasos WHERE
Producto LIKE '" & comp & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            Caso(0) = oDataReader("Familia")
            Caso(1) = oDataReader("NoElementos")
            Caso(2) = oDataReader("Geometria")
            Caso(3) = oDataReader("TotalOp")
            Caso(4) = oDataReader("TipoDeLamina")
            Caso(5) = oDataReader("Calibre")
            Caso(6) = oDataReader("TipoDeProducto")
            Caso(7) = oDataReader("Ancho")
            Caso(8) = oDataReader("Largo")
            Caso(9) = oDataReader("Profundidad")
```

```
            Caso(10) = oDataReader("Diametro")
            Caso(11) = oDataReader("Capacidad")

        End While
        oDataReader.Close()
        oConexion.Close()
    End Sub
    Sub comparacion2()
        Dim ven As New Resultado
        If Caso(0) = Elementos(0) Then
            ven.sim(0) = "Si"
        Else
            ven.sim(0) = "No"
        End If
        If Caso(1) = Elementos(1) Then
            ven.sim(1) = "Si"
        Else
            ven.sim(1) = "No"
        End If
        If Caso(2) = Elementos(2) Then
            ven.sim(2) = "Si"
        Else
            ven.sim(2) = "No"
        End If
        If Caso(3) = Elementos(3) Then
            ven.sim(3) = "Si"
        Else
            ven.sim(3) = "No"
        End If
        If Caso(4) = Elementos(4) Then
            ven.sim(4) = "Si"
        Else
            ven.sim(4) = "No"
        End If
        If Caso(5) = Elementos(5) Then
            ven.sim(5) = "Si"
        Else
            ven.sim(5) = "No"
        End If
        If Caso(6) = Elementos(6) Then
            ven.sim(6) = "Si"
        Else
            ven.sim(6) = "No"
        End If
        If Caso(7) = Elementos(7) Then
            ven.sim(7) = "Si"
        ElseIf Elementos(7) > (Caso(7) + 5) Then
            ven.sim(7) = "No"
        ElseIf Caso(7) < (Elementos(7) + 5) Then
            ven.sim(7) = "aproximado"
        ElseIf (Elementos(7) - 5) > Caso(7) Then
            ven.sim(7) = "No"
        End If
        If Caso(8) = Elementos(8) Then
            ven.sim(8) = "Si"
        ElseIf Elementos(8) > (Caso(8) + 5) Then
            ven.sim(8) = "No"
        ElseIf Caso(8) < (Elementos(8) + 5) Then
            ven.sim(8) = "aproximado"
        ElseIf (Elementos(8) - 5) > Caso(8) Then
            ven.sim(8) = "No"
        End If
        If Caso(9) = Elementos(9) Then
            ven.sim(9) = "Si"
        ElseIf Elementos(9) > (Caso(9) + 5) Then
            ven.sim(9) = "No"
        ElseIf Caso(9) < (Elementos(9) + 5) Then
            ven.sim(9) = "aproximado"
```

```
        ElseIf (Elementos(9) - 5) > Caso(9) Then
            ven.sim(9) = "No"
        End If
        If Caso(10) = Elementos(10) Then
            ven.sim(10) = "Si"
        ElseIf Elementos(10) > (Caso(10) + 5) Then
            ven.sim(10) = "No"
        ElseIf Caso(10) < (Elementos(10) + 5) Then
            ven.sim(10) = "aproximado"
        ElseIf (Elementos(9) - 5) > Caso(10) Then
            ven.sim(10) = "No"
        End If
        If Caso(11) = Elementos(11) Then
            ven.sim(11) = "Si"
        Else
            ven.sim(11) = "No"
        End If
    End Sub
    Sub productos()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM TipoDeProducto",
oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            Me.cbProd.Items.Add(oDataReader("TipoDeProducto"))

        End While
        oDataReader.Close()
        oConexion.Close()
    End Sub
End Class
```

Pantalla: Resultados

```
'Desplega los datos del caso obtenido de la comparación con el caso nuevo,
'describiendo los troqueles que se emplearón en su creación
Imports System.Data.OleDb
Imports System.Drawing.Printing

Public Class Resultado
    Inherits System.Windows.Forms.Form
    Public prod As String
    Public rb As New rbcN
    Public Caso(11) As String
    Public sim(11) As String
    Public c As Integer
    Private WithEvents DocImprimirTicket1 As New PrintDocument
    Private Sub ListBox1_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ListBox1.SelectedIndexChanged
        Me.ListBox5.Items.Clear()
        Me.ListBox6.Items.Clear()

        Me.Label6.Text = "Descripción del troquel " & Me.ListBox1.SelectedItem
        DescTroquel()
        llenarPartes()
        Try
            Dim temp As String = Me.ListBox1.SelectedItem
            PictureBox2.Image = Image.FromFile("C:\Imagenes\" & temp & ".jpg")
        Catch ex As Exception
            PictureBox2.Image = Image.FromFile("C:\Imagenes\SinImagen.jpg")
        End Try
```

```
    End Sub
    Sub llenarCheck()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim obComando As New OleDbCommand("SELECT * FROM Troquel WHERE
Producto LIKE '" & prod & "'", oConexion)

        ' crear DataReader
        Dim obDataReader As OleDbDataReader
        oConexion.Open()
        obDataReader = obComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While obDataReader.Read()
            'Me.ListBox1.Items.Add(oDataReader("Codigo"))
            Me.ListBox1.Items.Add(obDataReader("Codigo"))

        End While
        obDataReader.Close()
        oConexion.Close()
    End Sub
    Sub Descripción()
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM BaseDeCasos WHERE
Producto LIKE '" & prod & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()

            Me.ListBox2.Items.Add("Familia:  " & oDataReader("Familia"))
            Me.ListBox2.Items.Add("Componentes con los que se relaciona:  " &
oDataReader("NoElementos"))
            Me.ListBox2.Items.Add("Forma geometrica:  " &
oDataReader("Geometria"))
            Me.ListBox2.Items.Add("Operaciones:  " & oDataReader("TotalOp"))
            Me.ListBox2.Items.Add("Elaborado con lámina:  " &
oDataReader("TipoDeLamina"))
            Me.ListBox2.Items.Add("calibre:  " & oDataReader("Calibre"))
            Me.ListBox2.Items.Add("Tipo de producto:  " &
oDataReader("TipoDeProducto"))
            Me.ListBox2.Items.Add("Ancho:  " & oDataReader("Ancho") & " [mm]")
            Me.ListBox2.Items.Add("Largo:  " & oDataReader("Largo") & " [mm]")
            Me.ListBox2.Items.Add("Profundidad:  " &
oDataReader("Profundidad") & " [mm]")
            Me.ListBox2.Items.Add("Diámetro:  " & oDataReader("Diametro") & "
[mm]")
            Me.ListBox2.Items.Add("Capacidad:  " & oDataReader("Capacidad"))

            Caso(0) = oDataReader("Familia")
            Caso(1) = oDataReader("NoElementos")
            Caso(2) = oDataReader("Geometria")
            Caso(3) = oDataReader("TotalOp")
            Caso(4) = oDataReader("TipoDeLamina")
            Caso(5) = oDataReader("Calibre")
            Caso(6) = oDataReader("TipoDeProducto")
            Caso(7) = oDataReader("Ancho")
            Caso(8) = oDataReader("Largo")
            Caso(9) = oDataReader("Profundidad")
            Caso(10) = oDataReader("Diametro")
            Caso(11) = oDataReader("Capacidad")
```

```
        End While
        oDataReader.Close()
        oConexion.Close()
    End Sub
    Sub DescTroquel()
        'Dim ven As New Resultado
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM Troquel WHERE Codigo
LIKE '" & Me.ListBox1.SelectedItem & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()

            Me.ListBox5.Items.Add("Familia:  " & oDataReader("Familia"))
            Me.ListBox5.Items.Add("Realiza la operación " &
oDataReader("Operacion") & " de un total de " & oDataReader("OpDeFamilia"))
            Me.ListBox5.Items.Add("Tipo de operación " &
oDataReader("TipoDeOperacion"))
            Me.ListBox5.Items.Add("Forma geometrica:  " &
oDataReader("Geometria"))
            Me.ListBox5.Items.Add("Es troquel:  " &
oDataReader("TipoDeTroquel"))
            Me.ListBox5.Items.Add("Se ubíca en  " & oDataReader("Ubicacion"))
            Me.ListBox5.Items.Add("En: " & oDataReader("Localizacion"))
            Me.ListBox5.Items.Add("Uso:  " & oDataReader("ConstanciaDeUso"))
            Me.ListBox5.Items.Add("Observaciones:  " &
oDataReader("Observaciones"))

        End While
        oDataReader.Close()
        oConexion.Close()

    End Sub
    Sub llenarPartes()
        ' Dim ven As New Resultado
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM Partes WHERE Codigo
LIKE '" & Me.ListBox1.SelectedItem & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()
            Me.ListBox6.Items.Add("El troquel tiene " &
oDataReader("Cantidad") & oDataReader("Parte") & " de material " &
oDataReader("Material") & " en un estado " & oDataReader("EstadoFisico"))

        End While
        oDataReader.Close()
        oConexion.Close()

    End Sub


    Sub Relacion()
```

```
        Dim oConexion As New OleDbConnection()
        oConexion.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=BDGZ.mdb;"
        ' crear comando
        Dim oComando As New OleDbCommand("SELECT * FROM RelComponentes WHERE
Producto LIKE '" & prod & "'", oConexion)

        ' crear DataReader
        Dim oDataReader As OleDbDataReader
        oConexion.Open()
        oDataReader = oComando.ExecuteReader() ' obtener DataReader
        ' recorrer filas
        While oDataReader.Read()

            Me.ListBox4.Items.Add(oDataReader("Componente"))

        End While
        oDataReader.Close()
        oConexion.Close()
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)

    End Sub
'Creación del formato para imprimir los resultados de la búsqueda
    Private Sub DocImprimirTicket1_PrintPage(ByVal sender As Object, ByVal e
As System.Drawing.Printing.PrintPageEventArgs) Handles
DocImprimirTicket1.PrintPage
        Dim Fuente As Font
        Fuente = New Font("Arial", 10, FontStyle.Bold)
        Dim Fuente1 As Font
        Fuente1 = New Font("Arial", 14, FontStyle.Bold)
        e.Graphics.DrawImage(Me.PictureBox3.Image, 50, 40, 70, 70)
        e.Graphics.DrawString("Solución del caso ", Fuente1, Brushes.Green,
350, 50)
        e.Graphics.DrawString("-----------------------------------------------
-------------------------------------------------------------------------
", Fuente1, Brushes.Green, 50, 100)

        e.Graphics.DrawString(Me.TextBox1.Text, Fuente1, Brushes.Green, 50,
125)

        Dim y As Integer = 130
        Dim x As Integer
        e.Graphics.DrawImage(Me.PictureBox1.Image, 450, 200, 200, 200)
        For x = 0 To Me.ListBox2.Items.Count - 1
            y = y + 20
            e.Graphics.DrawString(Me.ListBox2.Items(x), Fuente,
Brushes.BlueViolet, 50, y)
        Next
        Dim z As Integer = 400
        e.Graphics.DrawString(Me.Label5.Text, Fuente1, Brushes.Green, 50, 400)
        For x = 0 To Me.ListBox1.Items.Count - 1
            z = z + 20
            e.Graphics.DrawString(Me.ListBox1.Items(x), Fuente, Brushes.Blue,
50, z)
        Next
        Dim w As Integer = z + 20
        Try
            e.Graphics.DrawImage(Me.PictureBox2.Image, 450, w + 20, 200, 200)

        Catch ex As Exception
            e.Graphics.DrawString("No hay imagen que mostrar", Fuente,
Brushes.Blue, 450, w + 20)
        End Try
        e.Graphics.DrawString(Me.Label6.Text, Fuente1, Brushes.Green, 50, z +
15)
```

```
        For x = 0 To Me.ListBox5.Items.Count - 1
            w = w + 20
            e.Graphics.DrawString(Me.ListBox5.Items(x), Fuente, Brushes.Blue,
50, w)
        Next
        Dim u As Integer = w + 20
        e.Graphics.DrawString(Me.Label7.Text, Fuente1, Brushes.Green, 50, w +
15)

        For x = 0 To Me.ListBox6.Items.Count - 1
            u = u + 20
            e.Graphics.DrawString(Me.ListBox6.Items(x), Fuente, Brushes.Blue,
50, u)
        Next
        Dim t As Integer = u + 20
        e.Graphics.DrawString(Me.Label4.Text, Fuente1, Brushes.Green, 50, u +
15)

        For x = 0 To Me.ListBox4.Items.Count - 1
            t = t + 20
            e.Graphics.DrawString(Me.ListBox4.Items(x), Fuente, Brushes.Blue,
50, t)
        Next

        e.HasMorePages = False
        Fuente.Dispose()
   End Sub

    Function PrepararTicket() As System.Drawing.Printing.PrintDocument

        Dim objPrintDocument As New System.Drawing.Printing.PrintDocument
        AddHandler objPrintDocument.PrintPage, AddressOf
DocImprimirTicket1_PrintPage
        Return objPrintDocument

    End Function

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)

    End Sub
    Sub comparacion2()

        If Caso(0) = rb.Elementos(0) Then
            sim(0) = "Si"
        Else
            sim(0) = "No"
        End If
        If Caso(1) = rb.Elementos(1) Then
            sim(1) = "Si"
        Else
            sim(1) = "No"
        End If
        If Caso(2) = rb.Elementos(2) Then
            sim(2) = "Si"
        Else
            sim(2) = "No"
        End If
        If Caso(3) = rb.Elementos(3) Then
            sim(3) = "Si"
        Else
            sim(3) = "No"
        End If
        If Caso(4) = rb.Elementos(4) Then
            sim(4) = "Si"
        Else
            sim(4) = "No"
```

```
        End If
        If Caso(5) = rb.Elementos(5) Then
            sim(5) = "Si"
        Else
            sim(5) = "No"
        End If
        If Caso(6) = rb.Elementos(6) Then
            sim(6) = "Si"
        Else
            sim(6) = "No"
        End If
        If Caso(7) = rb.Elementos(7) Then
            sim(7) = "Si"
        ElseIf rb.Elementos(7) > (Caso(7) + 5) Then
            sim(7) = "No"
        ElseIf Caso(7) < (rb.Elementos(7) + 5) Then
            sim(7) = "aproximado"
        ElseIf (rb.Elementos(7) - 5) > Caso(7) Then
            sim(7) = "No"
        End If
        If Caso(8) = rb.Elementos(8) Then
            sim(8) = "Si"
        ElseIf rb.Elementos(8) > (Caso(8) + 5) Then
            sim(8) = "No"
        ElseIf Caso(8) < (rb.Elementos(8) + 5) Then
            sim(8) = "aproximado"
        ElseIf (rb.Elementos(8) - 5) > Caso(8) Then
            sim(8) = "No"
        End If
        If Caso(9) = rb.Elementos(9) Then
            sim(9) = "Si"
        ElseIf rb.Elementos(9) > (Caso(9) + 5) Then
            sim(9) = "No"
        ElseIf Caso(9) < (rb.Elementos(9) + 5) Then
            sim(9) = "aproximado"
        ElseIf (rb.Elementos(9) - 5) > Caso(9) Then
            sim(9) = "No"
        End If
        If Caso(10) = rb.Elementos(10) Then
            sim(10) = "Si"
        ElseIf rb.Elementos(10) > (Caso(10) + 5) Then
            sim(10) = "No"
        ElseIf Caso(10) < (rb.Elementos(10) + 5) Then
            sim(10) = "aproximado"
        ElseIf (rb.Elementos(9) - 5) > Caso(10) Then
            sim(10) = "No"
        End If
        If Caso(11) = rb.Elementos(11) Then
            sim(11) = "Si"
        Else
            sim(11) = "No"
        End If
    End Sub



    Private Sub ToolStripMenuItem1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles ToolStripMenuItem1.Click
        DocImprimirTicket1.Print()
    End Sub

    Private Sub ToolStripMenuItem2_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles ToolStripMenuItem2.Click
        Me.PrintPreviewDialog1.Document = PrepararTicket()
        Me.PrintPreviewDialog1.WindowState = FormWindowState.Maximized
        Me.PrintPreviewDialog1.ShowDialog()
 End Sub
```

```
 End Class
```

Pantalla: Administrador

```
'Se introduce los datos de administrador y se verifican para el acceso a las
'aplicaciones

Imports System.Data.OleDb
Public Class LoginAdm
    Private oDataAdapter As OleDbDataAdapter
    Private oDataSet As DataSet
    Private oDataRow As DataRow
    Private ipos As Integer

    Private Sub OK_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles OK.Click
        Try
            'conexion a la base de datos
            Dim oConexion As OleDbConnection
            oConexion = New
OleDbConnection("provider=Microsoft.jet.oledb.4.0;Data source=BDGZ.mdb;")
            Me.oDataAdapter = New OleDbDataAdapter("select * from Usuarios
where idUsuario like '" & Me.UsernameTextBox.Text & "' and  Contraseña like '"
& Me.PasswordTextBox.Text & "' and Privilegio like '" & "Administrador" & "'",
oConexion)
            Me.oDataSet = New DataSet
            oConexion.Open()
            Me.oDataAdapter.Fill(oDataSet, "Usuarios")
            oConexion.Close()
            Me.ipos = 0
            Me.Carga()
            Me.Close()

            Me.UsernameTextBox.Clear()
            Me.PasswordTextBox.Clear()
        Catch ex As Exception
            MessageBox.Show("Usuario o Contraseña no validos", "",
MessageBoxButtons.OK)
            Me.PasswordTextBox.Clear()
            Me.UsernameTextBox.Clear()
        End Try
    End Sub

    Private Sub Cancel_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Cancel.Click
        Me.Close()
    End Sub
    Private Sub Carga()
        oDataRow = Me.oDataSet.Tables("Usuarios").Rows(Me.ipos)

        Dim vmenu As New MenuAdmin
        vmenu.Show()
    End Sub
End Class
```

Pantalla: administrador

```
'Se desplegan las diferentes aplicaciones en las que participara el
'responsable de la administración del sistema
Imports System.Data.OleDb
Public Class MenuAdmin
    Private oDataAdapter As OleDbDataAdapter
    Private oDataSet As DataSet
    Private ipos As Integer
    Private nom As String
    Private Sub toolAlta_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles toolAlta.Click
        Dim frmVentanaDentro As New AltaUsuarios
```

```
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Alta de usuario"
        frmVentanaDentro.Show()
    End Sub

    Private Sub ToolBaja_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolBaja.Click
        nom = InputBox("Dame el Nombre del usuario a dar de baja", "Baja de
usuario")


        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        Me.oDataAdapter = New OleDbDataAdapter("delete * from Usuarios WHERE
Nombre LIKE '" & nom & "'", oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(oDataAdapter)
        Me.oDataSet = New DataSet
        oConexion.Open()
        Me.oDataAdapter.Fill(oDataSet, "Usuarios")
        oConexion.Close()
        Me.ipos = 0
    End Sub

    Private Sub ToolPrestamo_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ToolPrestamo.Click
        Dim frmVentanaDentro As New Prestamo
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Prestamo de troqueles"
        frmVentanaDentro.Show()
    End Sub

    Private Sub RegistrarNuevaFamiliaToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RegistrarNuevaFamiliaToolStripMenuItem.Click
        Dim frmVentanaDentro As New Registros
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Agregar familia"
        frmVentanaDentro.Show()
    End Sub

    Private Sub RegistrarNuevaOperaciónToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RegistrarNuevaOperaciónToolStripMenuItem.Click
        Dim frmVentanaDentro As New AgregarOp
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Agregar operación"
        frmVentanaDentro.Show()
    End Sub

    Private Sub RegistrarPartePorOperaciónToolStripMenuItem_Click(ByVal sender
As System.Object, ByVal e As System.EventArgs) Handles
RegistrarPartePorOperaciónToolStripMenuItem.Click
        Dim frmVentanaDentro As New AgregarPxO
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Agregar parte de troquel por operación"
        frmVentanaDentro.Show()
    End Sub

    Private Sub RegistrarNuevaGeometríaToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RegistrarNuevaGeometríaToolStripMenuItem.Click
```

```
        Dim frmVentanaDentro As New AgregarGeo
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Agregar geometría"
        frmVentanaDentro.Show()
    End Sub

    Private Sub RegistrarNuevoTipoDeProductoToolStripMenuItem_Click(ByVal
sender As System.Object, ByVal e As System.EventArgs) Handles
RegistrarNuevoTipoDeProductoToolStripMenuItem.Click
        Dim frmVentanaDentro As New AgregarProducto
        Me.IsMdiContainer = True
        frmVentanaDentro.MdiParent = Me
        frmVentanaDentro.Text = "Agregar producto"
        frmVentanaDentro.Show()
    End Sub
End Class
```

Pantalla: alta de usuario

```
'Se registra un nuevo usuario proporcionando datos para permitir su ingreso a
'las aplicaciones, ya sea como usuario o como administrador
Imports System.Data.OleDb
Public Class AltaUsuarios
    Private oDataAdapter As OleDbDataAdapter
    Private oDataSet As DataSet
    Public oDataRow As DataRow
    Private objComando As New OleDb.OleDbCommand

    Private ipos As Integer
    Public nom As String
    Private Sub Alta_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        Me.oDataAdapter = New OleDbDataAdapter("select * from Usuarios",
oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(oDataAdapter)
        Me.oDataSet = New DataSet
        oConexion.Open()
        Me.oDataAdapter.Fill(oDataSet, "Usuarios")
        oConexion.Close()
        Me.iPos = 0
    End Sub
    Sub cargar()
        Dim oDataRow As DataRow
        oDataRow = Me.oDataSet.Tables("Usuarios").NewRow()
        oDataRow("idUsuario") = Me.txtusuario.Text
        oDataRow("Nombre") = Me.txtnombre.Text
        oDataRow("Contraseña") = Me.txtcontraseña.Text
        oDataRow("Privilegio") = Me.cbPriv.SelectedItem
        Me.oDataSet.Tables("Usuarios").Rows.Add(oDataRow)
        Me.oDataAdapter.Update(Me.oDataSet, "Usuarios")

    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Try
            Dim x, y As String
            x = Me.txtcontraseña.Text
            y = Me.TextBox3.Text
            If x = y Then

                If MessageBox.Show("¿Desea guardar el registro del nuevo
usuario?", "Atención", MessageBoxButtons.YesNoCancel, _
                MessageBoxIcon.Question) = Windows.Forms.DialogResult.Yes Then
```

```
                    cargar()
                    txtusuario.Clear()
                    txtnombre.Clear()
                    txtcontraseña.Clear()
                    Me.TextBox3.Clear()

                End If
            Else
                MessageBox.Show("No coincide la contraseña, por favor ingrese
nuevamente ", "", MessageBoxButtons.OK)
                Me.TextBox3.Clear()
                Me.txtcontraseña.Clear()
            End If
        Catch ex As Exception
            MessageBox.Show("Este usuario ya existe y tiene una contraseña",
"Atención", MessageBoxButtons.OK)
            Me.TextBox3.Clear()
            Me.txtcontraseña.Clear()
            Me.txtnombre.Clear()
            Me.txtusuario.Clear()

        End Try
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Me.txtnombre.Clear()
        Me.txtusuario.Clear()
        Me.txtcontraseña.Clear()
        Me.TextBox3.Clear()
    End Sub

    Private Sub BajaDeToolStripMenuItem_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs)

    End Sub

    Private Sub PrestamoDeTroquelesToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs)
        Dim vPrestamo As New Prestamo
        vPrestamo.Show()
    End Sub
End Class
```

Pantalla: Prestamo de troqueles

```
'Se lleva el registro de los troqueles que están prestados a las distintas
'áreas de la planta
Imports System.Data.OleDb
Public Class Prestamo
    Private oDataAdapter, daT As OleDbDataAdapter
    Private oDataSet, dsT As DataSet
    Public oDataRow, drT As DataRow
    Private ipos As Integer
    Private devol, nom As String
    Dim MiFecha As Date
    Dim troq(12) As String
    Private Sub USUARIOS_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        oDataAdapter = New OleDbDataAdapter("SELECT * FROM Prestamo",
oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New
OleDbCommandBuilder(oDataAdapter)
        Me.oDataSet = New DataSet
        oConexion.Open()
```

```
        Me.oDataAdapter.Fill(oDataSet, "Prestamo")
        oConexion.Close()
        Me.DataGridView1.DataSource = oDataSet
        Me.DataGridView1.DataMember = "Prestamo"
        ipos = 0

    End Sub
    Sub cargar()
        Try
            drT = dsT.Tables("Troquel").Rows(Me.ipos)
            troq(0) = drT("Codigo")
            troq(1) = drT("Familia")
            troq(2) = drT("OpDeFamilia")
            troq(3) = drT("Operacion")
            troq(4) = drT("TipoDeOperacion")
            troq(5) = drT("Geometria")
            troq(6) = drT("TipoDeTroquel")
            troq(7) = drT("Producto")
            troq(8) = drT("Ubicacion")
            troq(9) = drT("Localizacion")
            troq(10) = drT("ConstanciaDeUso")
            troq(11) = drT("Observaciones")
            troq(12) = drT("Imagen")
            Me.ipos = (Me.dsT.Tables("Troquel").Rows.Count - 1)
        Catch ex As Exception
            MessageBox.Show("Este código no existe")
        End Try


    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
        MiFecha = Now() ' #1/19/2002 12:27:08 PM#
        Label10.Text = MiFecha
    End Sub

    Private Sub btncancelar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Me.txtsolicitante.Clear()
        Me.txtarea.Clear()
        Me.txtcodigo.Clear()
        Me.txtsolicitante.Clear()
    End Sub
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Me.oDataAdapter.Update(oDataSet, "Prestamo")
        MessageBox.Show("Devolución guardada")

    End Sub
    Private Sub btncancelar_Click_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btncancelar.Click
        Me.Close()
    End Sub

    Private Sub btnguardar_Click_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnguardar.Click
        Dim oDataRow As DataRow
        oDataRow = Me.oDataSet.Tables("Prestamo").NewRow()
        oDataRow("Codigo") = Me.txtcodigo.Text
        oDataRow("Solicitante") = Me.txtsolicitante.Text
        oDataRow("FechaSalida") = Me.Label10.Text
        oDataRow("FechaEntrega") = #1/1/2007 12:00:00 PM#
        Me.oDataSet.Tables("Prestamo").Rows.Add(oDataRow)

        Me.oDataAdapter.Update(Me.oDataSet, "Prestamo")

        Me.guardarTroquel()
```

```
        MessageBox.Show("Registro guardado en la base de datos")
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        If MessageBox.Show("¿Está seguro de ELIMINAR este registro?",
"Eliminar Registro", MessageBoxButtons.YesNo, MessageBoxIcon.Question) =
Windows.Forms.DialogResult.Yes Then
            Dim obDataRow As DataRow

            Dim rowy As Integer
            rowy = Me.DataGridView1.CurrentCellAddress.Y
            obDataRow = Me.oDataSet.Tables("Prestamo").Rows(rowy)
            obDataRow.Delete()
            Dim oTablaBorrados As DataTable
            oTablaBorrados = _
            Me.oDataSet.Tables("Prestamo").GetChanges(DataRowState.Deleted)
            Me.oDataAdapter.Update(oTablaBorrados)
            Me.oDataSet.Tables("Prestamo").AcceptChanges()
        End If
    End Sub
    Sub TroquelM()
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        daT = New OleDbDataAdapter("SELECT * FROM Troquel where Codigo like '"
& Me.txtcodigo.Text & "'", oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(daT)
        Me.dsT = New DataSet
        oConexion.Open()
        Me.daT.Fill(dsT, "Troquel")
        oConexion.Close()

        ipos = 0
        cargar()
    End Sub
    Private Sub txtcodigo_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtcodigo.Leave
        Me.TroquelM()
    End Sub
    Sub guardarTroquel()

        drT = Me.dsT.Tables("Troquel").Rows(Me.ipos)
        drT("Codigo") = Me.txtcodigo.Text
        drT("Familia") = troq(1)
        drT("OpDeFamilia") = troq(2)
        drT("Operacion") = troq(3)
        drT("TipoDeOperacion") = troq(4)
        drT("Geometria") = troq(5)
        drT("TipoDeTroquel") = troq(6)
        drT("Producto") = troq(7)
        drT("Ubicacion") = Me.cbUbic.SelectedItem
        drT("Localizacion") = Me.txtarea.Text
        drT("ConstanciaDeUso") = troq(10)
        drT("Observaciones") = troq(11)
        drT("Imagen") = troq(12)
        Me.daT.Update(Me.dsT, "Troquel")
End Sub
    Private Sub Button1_Click_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim ven As New Devolución
        ven.Show()
    End Sub
End Class
```

Pantalla: Devolución de troqueles

```
'Se registra la devolución de los troqueles y modifica los datos que
'determinan su ubicación original
Imports System.Data.OleDb
Public Class Devolución
    Public daT, daP As OleDbDataAdapter
    Public dsT, dsP As DataSet
    Public drT, drP As DataRow
    Public ipos As Integer
    Public troq(12) As String
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Me.Close()
    End Sub


    Sub Devolucion()
      Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        Me.daP = New OleDbDataAdapter("select * from Prestamo WHERE Codigo
LIKE '" & Me.txtCodDEv.Text & "'", oConexion)
        Me.daT = New OleDbDataAdapter("select * from Troquel WHERE Codigo LIKE
'" & Me.txtCodDEv.Text & "'", oConexion)

        Dim oCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(daP)
        Dim tCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(daT)

        Me.dsP = New DataSet
        Me.dsT = New DataSet

        oConexion.Open()
        Me.daP.Fill(dsP, "Prestamo")
        Me.daT.Fill(dsT, "Troquel")
        oConexion.Close()
        Me.ipos = 0

        cargar()
    End Sub
    Sub DevolucionTro()
        drT = Me.dsT.Tables("Troquel").Rows(Me.ipos)
        drT("Ubicacion") = Me.cbUbicDev.SelectedItem
        drT("Localizacion") = Me.txtLocDev.Text
        Me.daT.Update(Me.dsT, "Troquel")
    End Sub
    Sub guardarDev()
        drP = Me.dsP.Tables("Prestamo").Rows(Me.ipos)
        drP("FechaEntrega") = Now()
        Me.daP.Update(Me.dsP, "Prestamo")
    End Sub

    Private Sub btnAceptar_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAceptar.Click
        DevolucionTro()
        guardarDev()
        'Actualizar()
        Me.Close()

    End Sub
    Sub cargar()
        Try
            drT = dsT.Tables("Troquel").Rows(Me.ipos)
            troq(0) = drT("Codigo")
            troq(1) = drT("Familia")
            troq(2) = drT("OpDeFamilia")
            troq(3) = drT("Operacion")
            troq(4) = drT("TipoDeOperacion")
```

```
            troq(5) = drT("Geometria")
            troq(6) = drT("TipoDeTroquel")
            troq(7) = drT("Producto")
            troq(8) = drT("Ubicacion")
            troq(9) = drT("Localizacion")
            troq(10) = drT("ConstanciaDeUso")
            troq(11) = drT("Observaciones")
            troq(12) = drT("Imagen")
            Me.ipos = (Me.dsT.Tables("Troquel").Rows.Count - 1)

        Catch ex As Exception

        End Try
    End Sub

    Private Sub txtLocDev_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles txtLocDev.Leave
        Devolucion()
    End Sub

    Private Sub btnAceptar_Leave(ByVal sender As Object, ByVal e As
System.EventArgs) Handles btnAceptar.Leave
        Actualizar()
    End Sub
    Sub Actualizar()
        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        daP = New OleDbDataAdapter("SELECT * FROM Prestamo", oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(daP)
        Me.dsP = New DataSet
        oConexion.Open()
        Me.daP.Fill(dsP, "Prestamo")
        oConexion.Close()
        Dim vp As New Prestamo
        vp.DataGridView1.DataSource = dsP
        vp.DataGridView1.DataMember = "Prestamo"
        ipos = 0

    End Sub

    Private Sub Devolución_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub
End Class
```

Pantalla: Agregar Geometrías

```
'Se agregan datos nuevos o se quitan de la tabla Geometria
Imports System.Data.OleDb
Public Class AgregarGeo
    Private ds As DataSet
    Private da As OleDbDataAdapter
    Private dr As DataRow
    Private i As Integer
    Private Sub AgregarGeo_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim cn As OleDbConnection
        cn = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        da = New OleDbDataAdapter(" select * from Geometria", cn)
        Dim cb As OleDbCommandBuilder = New OleDbCommandBuilder(da)
        Me.ds = New DataSet
        cn.Open()
        da.Fill(ds, "Geometria")
        cn.Close()
        Me.DataGridView1.DataSource = ds
        Me.DataGridView1.DataMember = "Geometria"
```

```
        Me.i = 0
        Me.DataGridView1.Columns(1).Width = 300
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim idg, geo As String
        idg = InputBox("Identificador de la geometría", " ", "")
        geo = InputBox("Geometría", "", "")


        dr = Me.ds.Tables("Geometria").NewRow()
        dr("idGeometria") = idg
        dr("Geometria") = geo
        Try
            Me.ds.Tables("Geometria").Rows.Add(dr)
            Me.da.Update(Me.ds, "Geometria")
        Catch ex As Exception
            MessageBox.Show("Esta geometría ya existe")
        End Try

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click

        Dim nomgeo As String
        nomgeo = InputBox("Dame el Nombre de la geometría a dar de baja",
"Quitar geometría")


        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        Me.da = New OleDbDataAdapter("delete * from Geometria WHERE Geometria
LIKE '" & nomgeo & "'", oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(da)
        Me.ds = New DataSet
        oConexion.Open()
        Me.da.Fill(ds, "Geometria")
        oConexion.Close()
        Me.i = 0
        Try
        Catch ex As Exception

        End Try



    End Sub
End Class
```

Pantalla: Agregar Operación

```
'Se agregan datos nuevos o se quitan de la tabla Operación

Public Class AgregarOp
    Public ds As DataSet
    Public da As OleDbDataAdapter
    Public dr As DataRow
    Public i As Integer
    Private Sub AgregarOp_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim cn As OleDbConnection
        cn = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        da = New OleDbDataAdapter(" select * from TipoDeOperacion", cn)
        Dim cb As OleDbCommandBuilder = New OleDbCommandBuilder(da)
        Me.ds = New DataSet
```

```
        cn.Open()
        da.Fill(ds, "TipoDeOperacion")
        cn.Close()
        Me.DataGridView1.DataSource = ds
        Me.DataGridView1.DataMember = "TipoDeOperacion"
        Me.i = 0
        Me.DataGridView1.Columns(1).Width = 300
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Try
            Dim idop, operación As String
            idop = InputBox("Identificador de la operación", " ", "")
            operación = InputBox("Operación", "", "")


            dr = Me.ds.Tables("TipoDeOperacion").NewRow()
            dr("idOperacion") = idop
            dr("TipoDeOperacion") = operación

            Me.ds.Tables("TipoDeOperacion").Rows.Add(dr)
            Me.da.Update(Me.ds, "TipoDeOperacion")
        Catch ex As Exception
            MessageBox.Show("Nombre de la operación duplicado o nulo")
        End Try

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim nom As String
        nom = InputBox("Dame el Nombre de la operación a dar de baja", "Quitar
operación")


        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        Me.da = New OleDbDataAdapter("delete * from TipoDeOperacion WHERE
TipoDeOperacion LIKE '" & nom & "'", oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(da)
        Me.ds = New DataSet
        oConexion.Open()
        Me.da.Fill(ds, "TipoDeOperacion")
        oConexion.Close()
        Me.i = 0

    End Sub
End Class


Pantalla Agregar Producto

Imports System.Data.OleDb
Public Class AgregarProducto
    Public ds As DataSet
    Public da As OleDbDataAdapter
    Public dr As DataRow
    Public i As Integer
    Private Sub AgregarProducto_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim cn As OleDbConnection
        cn = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        da = New OleDbDataAdapter(" select * from TipoDeProducto", cn)
        Dim cb As OleDbCommandBuilder = New OleDbCommandBuilder(da)
        Me.ds = New DataSet
```

```
        cn.Open()
        da.Fill(ds, "TipoDeProducto")
        cn.Close()
        Me.DataGridView1.DataSource = ds
        Me.DataGridView1.DataMember = "TipoDeProducto"
        Me.i = 0
        Me.DataGridView1.Columns(0).Visible = False
        Me.DataGridView1.Columns(1).Width = 350
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Try
            Dim producto As String

            producto = InputBox("Tipo de producto", "", "")


            dr = Me.ds.Tables("TipoDeProducto").NewRow()

            dr("TipoDeProducto") = producto

            Me.ds.Tables("TipoDeProducto").Rows.Add(dr)
            Me.da.Update(Me.ds, "TipoDeProducto")
        Catch ex As Exception
            MessageBox.Show("Nombre de la operación duplicado o nulo")
        End Try
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim nom As String
        nom = InputBox("Dame el Nombre del producto a dar de baja", "Quitar
producto")


        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        Me.da = New OleDbDataAdapter("delete * from TipoDeProducto WHERE
TipoDeProducto LIKE '" & nom & "'", oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(da)
        Me.ds = New DataSet
        oConexion.Open()
        Me.da.Fill(ds, "TipoDeProducto")
        oConexion.Close()
        Me.i = 0

    End Sub
End Class
```

Pantalla: Agregar Partes

```
'Se agregan datos nuevos o se quitan de la tabla PartesxOP

Imports System.Data.OleDb
Public Class AgregarPxO
    Private ds As DataSet
    Private da As OleDbDataAdapter
    Private dr As DataRow
    Private i As Integer

    Private Sub AgregarPxO_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim cn As OleDbConnection
        cn = New OleDbConnection("Provider=Microsoft.Jet.OleDB.4.0;Data
Source=BDGZ.mdb;")
        da = New OleDbDataAdapter(" select * from PartesxOp", cn)
        Dim cb As OleDbCommandBuilder = New OleDbCommandBuilder(da)
```

```
        Me.ds = New DataSet
        cn.Open()
        da.Fill(ds, "PartesxOp")
        cn.Close()
        Me.DataGridView1.DataSource = ds
        Me.DataGridView1.DataMember = "PartesxOp"
        Me.i = 0
        Me.DataGridView1.Columns(1).Width = 300
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Try
            Dim idxop, partexop As String
            idxop = InputBox("Identificador de la operación", " ", "")
            partexop = InputBox("Parte", "", "")


            dr = Me.ds.Tables("PartesxOp").NewRow()
            dr("idOperacion") = idxop
            dr("Parte") = partexop

            Me.ds.Tables("PartesxOp").Rows.Add(dr)
            Me.da.Update(Me.ds, "PartesxOp")
        Catch ex As Exception
            MessageBox.Show("las partes deben estar relacionadas con un tipo
de operación")
            Me.Close()
        End Try

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim nomp, idp As String
        nomp = InputBox("Dame el Nombre de la parte a dar de baja", "Quitar
parte")
        idp = InputBox("Dame el identificador de la operación", "Quitar
operador")

        Dim oConexion As OleDbConnection
        oConexion = New OleDbConnection("Provider=Microsoft.Jet.OleDb.4.0;Data
Source=BDGZ.mdb;")
        Me.da = New OleDbDataAdapter("delete * from PartesxOp WHERE Parte LIKE
'" & nomp & "' and idOperacion like '" & idp & "'", oConexion)
        Dim oCommBuild As OleDbCommandBuilder = New OleDbCommandBuilder(da)
        Me.ds = New DataSet
        oConexion.Open()
        Me.da.Fill(ds, "PartesxOp")
        oConexion.Close()
        Me.i = 0

    End Sub
End Class
```

Bibliografía

Autor: Blanco, Luís Miguel
Titulo: Programación en Visual Basic .Net
Editorial: Grupo Eidos, Madrid 2002

Autor: Jacobo Armendáriz, Víctor H.
Titulo: Sistema experto para análisis de falla de elementos mecánicos metálicos.
Año 2005

Autor: Jacobo Armendáriz, Víctor H. y Ramos Juárez, Efraín
Titulo: Reutilización de troqueles, su clasificación en familias y estudio para implementar la administración del ciclo de vida
Proyecto UDIATEM – Grupo Zapata, reportes.

Autor: Lauréenle, Doyle
Titulo: Materiales y procesos de manufactura para ingenieros
Editorial: Prentice Hall México 1988

Autor: López Navarro, Tomás
Titulo: Troquelado y Estampación con aplicaciones al punzonado, doblado embutición y extrusión.
Editorial: Gustavo Pili, S.A., Barcelona, 1976

Autor: Poole, David
Titulo: Computational intelligence a logical approach.
Editorial: Oxford University Press, Inc., USA 1998

Autor: Pressman, Roger S
Titulo: Ingeniería de software.
Editorial: McGraw-Hill, USA, 1997.

Autor: Rico Mora, José Antonio
Titulo: Ingeniería de manufactura
Editorial: Compañía editorial continental, S.A. de CV., México 1984

Autor: Stevens, Perdita
Titulo: Utilización de UML en ingeniería del software con objetos y componentes.
Editorial: Pearson educación, S.A., Madrid, 2002

Autor: Turban, Efraim
Titulo: Expert systems and applied artificial intelligence.
Editorial: Macmillan Publishing Company, New York 1992

Referencias en Internet

http://es.wikipedia.org/wiki/Razonamiento_basado_en_casos
Tema: Razonamiento basado en casos

http://www.gurugames.com.es/people/pedro/aad/intro_cbr.pdf
Tema: Metodología del razonamiento basado en casos

http://www.gzapata.com.mx
Tema: Descripción de la empresa

http://www.iie.org.mx/gee/arti/21.pdf
Tema: Razonamiento basado en casos
Articulo: Aplicación del rbc al diagnóstico de generadores eléctricos

http://www.ingenieria.uady.mx/weblioteca/sistemasinteligentes/Tema05/CBR1.htm
Tema: Razonamiento basado en casos